Visual Attention Models for Far-Field Scene Analysis

by

Tomáš Ižo

B.S., Yale University (2001)
S.M., MIT (2003)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2007

Author .....
Department of Electrical Engineering and Computer Science
May 10, 2007

Certified by .....
W. Eric L. Grimson
Bernard Gordon Professor of Medical Engineering
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Visual Attention Models for Far-Field Scene Analysis

by

Tomáš Ižo

Submitted to the Department of Electrical Engineering and Computer Science
on May 10, 2007, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science

Abstract

The amount of information available to an intelligent monitoring system is simply too vast to process in its entirety. One way to address this issue is by developing attentive mechanisms that recognize parts of the input as more interesting than others. We apply this concept to the domain of far-field activity analysis by addressing the problem of determining where to look in a scene in order to capture interesting activity in progress.

We pose the problem of attention as an unsupervised learning problem, in which the task is to learn from long-term observation a model of the usual pattern of activity. Such a statistical scene model then makes it possible to detect and attend to examples of unusual activity.

We present two data-driven scene modeling approaches. In the first, we model the pattern of individual observations (instances) of moving objects at each scene location as a mixture of Gaussians. In the second approach, we model the pattern of sequences of observations—tracks—by grouping them into clusters. We employ a similarity measure that combines comparisons of multiple attributes—such as size, position, and velocity—in a principled manner so that only tracks that are spatially similar and have similar attributes at spatially corresponding points are grouped together. We group the tracks using spectral clustering and represent the scene model as a mixture of Gaussians in the spectral embedding space. New examples of activity can be efficiently classified by projection into the embedding space.

We demonstrate clustering and unusual activity detection results on a week of activity in the scene (about 40,000 moving object tracks) and show that human perceptual judgments of unusual activity are well-correlated with the statistical model. The human validation suggests that the track-based anomaly detection framework would perform well as a classifier for unusual events. To our knowledge, our work is the first to evaluate a statistical scene modeling and anomaly detection framework against human judgments.

Thesis Supervisor: W. Eric L. Grimson
Title: Bernard Gordon Professor of Medical Engineering

Acknowledgments

Ked' som bol žiakom na základnej škole, v predrevolučnom Československu, moji rodičia mali úžasnú predvídavost'. Našli mi učitel'ku anglického jazyka—pre prípad, že by sa nám niekedy podarilo zbavit' sa železnej záclony. Neskôr, ked' som prejavil záujem o štúdium za oceánom, bezvýhradne ma podporili napriek vedomiu, že svoj dospelý život začnem tisícky kilometrov od nich a od celej mojej rodiny. Ťažko je nájst' slová, ktoré by dostatočne vyjadrili moju vd'aku za roky odriekania, podpory a lásky, na ktorú som sa vždy mohol spol'ahnút'. Mami, tati, túto diplomovú prácu s láskou venujem Vám.

- Tomáš

When I was attending elementary school in pre-Velvet Revolution Czechoslovakia, my parents had the brilliant foresight to encourage me to learn English—just in case we should eventually emerge from behind the Iron Curtain. Later, when I showed interest in studying across the ocean, they supported me unconditionally, despite knowing that I would begin my adult life thousands of miles away from them and the rest of my family. Needless to say, no amount of thanks will ever be adequate for the selfless support, encouragement and sacrifice that I have always received from my parents. Mom and Dad, I dedicate this dissertation to you, with love.

There are many people without whom I might have taken a different path through life. I am grateful to my Yale mentor, Steve Zucker, for introducing me to vision, both biological and computational, and for his kind words of encouragement and support. Many thanks to my MIT mentor, Eric Grimson, who has always made sure that I was free to pursue ideas that I found interesting, all the while guiding me in the right direction towards this dissertation. Eric's attention to detail and his dedication to his students, despite the burden of running the department, has been extraordinary, and his support, especially during times when I felt less sure of myself, has been invaluable.

I could not have found more helpful and supportive committee members—Leslie Kaelbling and Pawan Sinha. Leslie, thanks for your guidance and for the chance to help teach one of

my favorite courses at MIT—6.825. Pawan, thanks for your help with my first foray into cognitive science. Thank you both for your generous and kind feedback on this document.

I have been fortunate to have an incredible group of friends and colleagues, who have helped to keep me sane and happy over the six years I spent at MIT.

Kathy, thanks for being a perpetual source of strength for everyone around you, including me. I had a wonderful time living with you (and Fermi) for the first 5 years in Boston, and I will miss you both.

Josh, you started as my office mate in NE43 and have become one of my dearest friends. Thanks for always being there for me, both on campus and off. I could not have done this without you.

Lauren, thank you for the conversation that inspired one of the main ideas of this thesis and for the many helpful conversations that followed. Thank you also for always making me laugh.

I could always depend on the members of our research group, whether I needed a sounding board for ideas, feedback on my defense talk or on this document, or just a late night snack. Josh, Kinh, Chris, Gerald, Bis, Ulas, Xiaoxu, Chaowei, Xiaogang and Jenny—thanks! Thanks also to Ben Balas for his help with designing the user study.

Thanks to all of my MITOC friends for many fun weekends in the mountains.

Thanks to everyone in my family, especially my grandparents, for standing by me.

Finally, thanks to Rob, who has been a source of limitless support over the past two years. You have been a role model, a copy editor, a colleague, partner in life and adventure, and I could not have done this without your patience and love.

- Tomáš

Contents

List of Figures

List of Algorithms

List of Tables

Chapter 1

Introduction

In the early years of computer vision, when cameras were few and lacking in resolution, one of the central questions was that of recognition: "What are we looking at?" When machine vision began to play an important role in robotics, an additional question arose—the question of where to pay attention: "Where should we be looking?" Recently, as cameras have become more readily available and the amount of visual data to be analyzed has rapidly increased, this question has become even more critical. For people, determining where to look—where the "action" is—is quite easy. At the lowest level, the decision of where to move our eyes is made by neural circuitry in the brain, much of which is in place from birth. However, more complex decisions, such as where to attend in a video, require a great deal of experience in observing the world.

In this dissertation we address the problem of determining from long-term observation where to look in an active scene in order to capture interesting events while they are in progress. We approach the problem from the point of view of an automatic visual monitoring system, in which the goal is to direct attention to the most interesting activity happening in the scene.

Figure 1-1: The tradeoff between coverage and resolution. If the parts of the scene where motion occurs were to be covered at the resolution shown in the lower right (c), nearly 50 cameras would have to be tiled (25 are shown in a). If a single camera is used, only low resolution can be achieved. (b) and (c) show portions of the area in the white rectangle in a single camera vs. the tiled camera array setup, respectively.

1.1 The Role of Attention

To illustrate why attentive monitoring systems are needed, consider a camera observing a scene rich in various types of activity such as the parking lot in Figure 1-1. Analyzing the entire scene in great detail would require an enormous amount of resources. It might be better to instead perform a coarse analysis, which can be subsequently used to direct the attention of the system to an area of interest for further analysis. In a security setting, this might involve selecting the view of the interior or exterior of a building that contains a suspicious activity in progress. In the setting of a television broadcast of a sports event, it might involve selecting the view that contains a foul or the scoring of a point. In a home monitoring setting, it could mean selecting the view that contains an elderly person having

health difficulties. The common denominator of all of these tasks is the selection of the most visually interesting part of the scene as a target for further attention, potentially at a higher resolution.

To facilitate this further analysis, it is necessary to obtain images of sufficient detail, i.e. sufficient resolution. In far- and mid-field settings, there is typically a tradeoff between area of coverage—the area around a building, for instance, that comprises the visual field of a particular camera setup—and the available resolution, i.e. the level of detail available for analysis. Figure 1-1 illustrates the tradeoff. If only one camera with the resolution of 640x480 pixels were used to cover the entire area of the shown parking lot, there would not be enough detail to analyze activity in the scene: notice that in part (b) of the figure, the person passing in front of the car is barely discernable as compared to the high-resolution image shown in part (c). To be able to obtain video from the entire parking lot area at 10x resolution, more than fifty cameras at 640x480 pixels would have to be tiled. This number would be even larger if the cameras were tiled with an overlap in order to facilitate the tracking of moving objects. On the other hand, if only one camera zoomed to 10x resolution were used, the area of coverage of the scene would be extremely small.

An efficient solution to the problem of having to compromise between coverage and resolution can be achieved by using a focus-of-attention camera system. Such systems have recently gained popularity and were originally inspired by the design of the mammalian visual system. The fovea of the mammalian eye—the central, high acuity area of the retina—has a much higher concentration of photosensitive cells, the photoreceptors, than the peripheral regions. Whereas the foveal area of the human retina contains around 200,000 photoreceptors per square milimeter, the number is ten times smaller five degrees out from the center of the fovea and one hundred times smaller ten degrees out [44]. If the entire retina had the same density of photoreceptors as the foveal region, the size of the optic nerve exiting the retina would be biologically prohibitive. However, because only a tiny area of the eye is capable of high acuity, the eye has to move so that the observed object or part of the scene falls onto the fovea. Thus the problem of visual attention arises. Similarly, a focus-of-

attention camera system usually consists of a stationary, master camera with a wide angle of view and a taskable, pan-tilt-zoom (PTZ), high-resolution camera. In order for the high-resolution camera to obtain images of interesting activity in the scene, an appropriate model of attention for the system is needed. In this dissertation, we describe how such a model can be built by learning the usual pattern of activity in the scene from long-term observation.

1.2 Attention as a Learning Problem

In order to better define the problem, we need to make some assumptions. First, how do we define an activity? For the purposes of this dissertation, we will consider an activity to be the collection of observations associated with the path of a single moving object through the scene. An example is shown in Figure 1-4(a). As we will see in the following chapter, other definitions have been proposed. We use the path of a single moving object partly because it lends itself well to statistical analysis and also because it is a natural building block for more complex activities, such as interactions between multiple objects.

Second, what do we consider interesting? In the human sense, the word "interesting" applies to anything to which a person might choose to pay attention. It could be something suspicious, funny, unexpected, unusual, surprising or anything else that might catch one's eye. Recognizing events to which a person might ascribe subjective terms such as "suspicious" or "amusing" is beyond the capabilities of current artificial intelligence algorithms. However, we *do* have the ability to recognize events that are unusual or unexpected if we consider those terms in the statistical sense. By modeling the usual pattern of activity in the scene, we can determine how statistically unusual an event is by comparing it to the model. The problem of attention thus becomes a learning problem.

1.3 Learning a Scene Model

Having defined what we will consider interesting, our approach to detecting unusual activity is to learn a statistical model of the normal range of activity in the scene. As in any learning

problem, the more data we use for the learning, the more descriptive the model will be. A day of observation of a typical scene may contain tens of thousands of moving objects of various sizes (people, groups, bicycles, motorbikes, cars, trucks, etc.) taking a wide variety of different paths through the scene. Consequently, dealing with a large amount of data becomes a central issue. We want to distill, from this huge volume of data, compact models of the common (and uncommon) activities in the scene without prior assumptions about those activities. We can then use these models to detect unusual activities even as they are occuring by flagging deviations from normal modes.

We describe two different approaches to building a statistical scene model. The first is based on individual observations of moving objects as they pass through the scene. Given such observations over a long period of time, we build an attribute map of the scene. For every image location, the attribute map describes the distribution of the object attributes, such as size, velocity, orientation, etc., that have been observed at that location (see Figure 1-2). This makes it possible to reason about the likelihood of new observations and detect ones that are unlikely given the model. Figure 1-2 shows an example unusual event: a person crossing an area where moving objects are less likely.

One natural limitation of an observation-based scene model is that it does not allow for the detection of events that consist of likely observations occuring in an unlikely sequence. We overcome this problem by presenting a second scene analysis approach that models the tracks of objects moving through the scene. Each track consists of an ordered set of observations containing attributes such as position, size, direction of motion, velocity, etc. We group moving object tracks using spectral clustering and model the scene as a mixture of Gaussians in the spectral embedding space. Our clustering method uses a multi-attribute distance measure, under which tracks appear similar only when the objects take a similar path through the scene *and* have similar attributes at spatially corresponding points along that path. For instance, a person and a car taking the same path through the scene or two cars traveling along the same path but at different speeds will not appear as similar. An example clustering result is shown in Figure 1-3. We compare new object paths to the

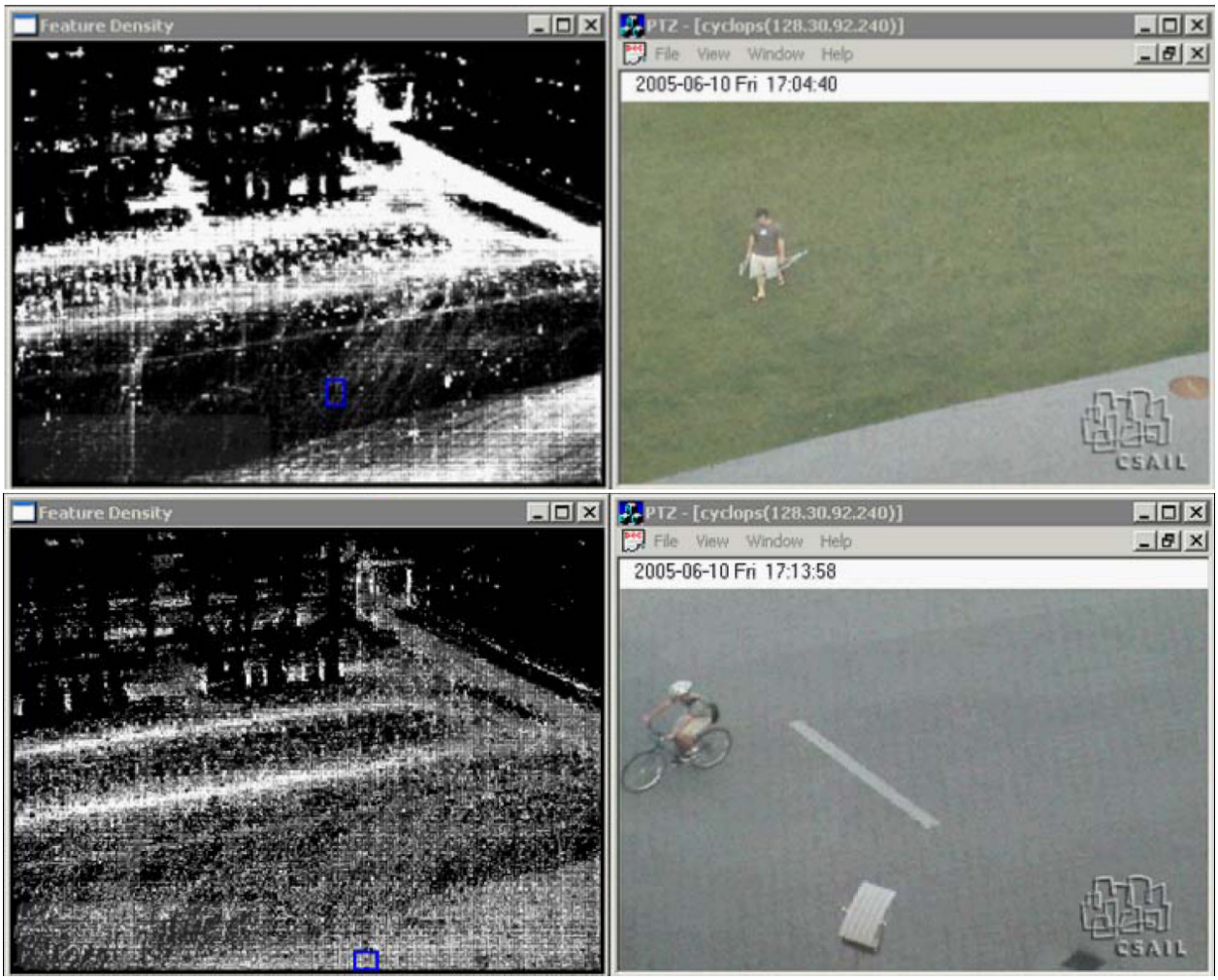
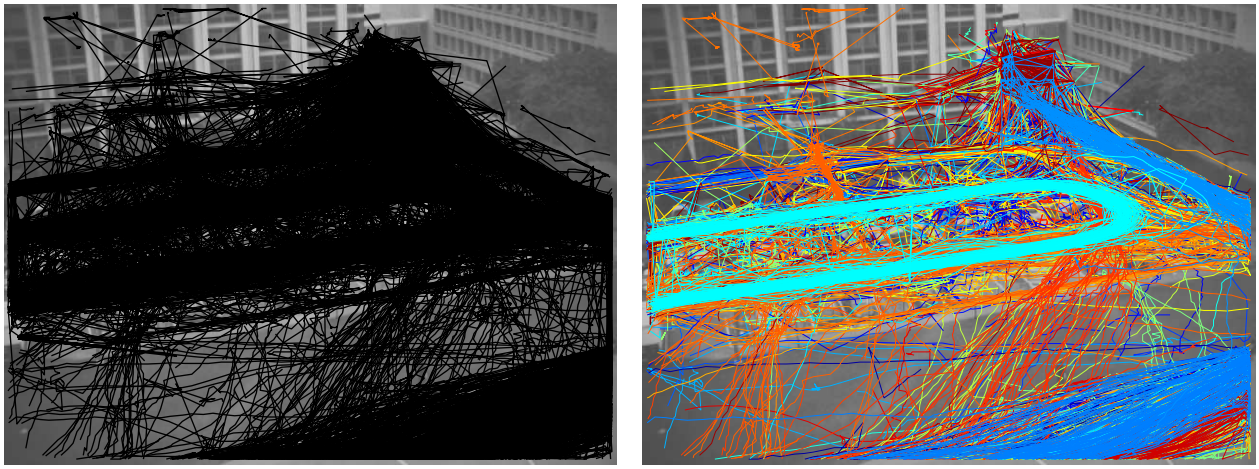


Figure 1-2: Top Left: attribute map showing at each pixel the likelihood of a moving object's centroid passing through that location; brighter pixels indicate higher likelihood. The blue rectangle marks an unlikely observation. Top Right: high resolution image of the marked location reveals a person crossing an area where moving objects are unlikely. Bottom Left: attribute map showing the mean speed of objects at every location. The blue rectangle marks an object with an unlikely speed. Bottom Right: high resolution image of the marked area reveals a person riding a bicycle in a region where people normally walk.



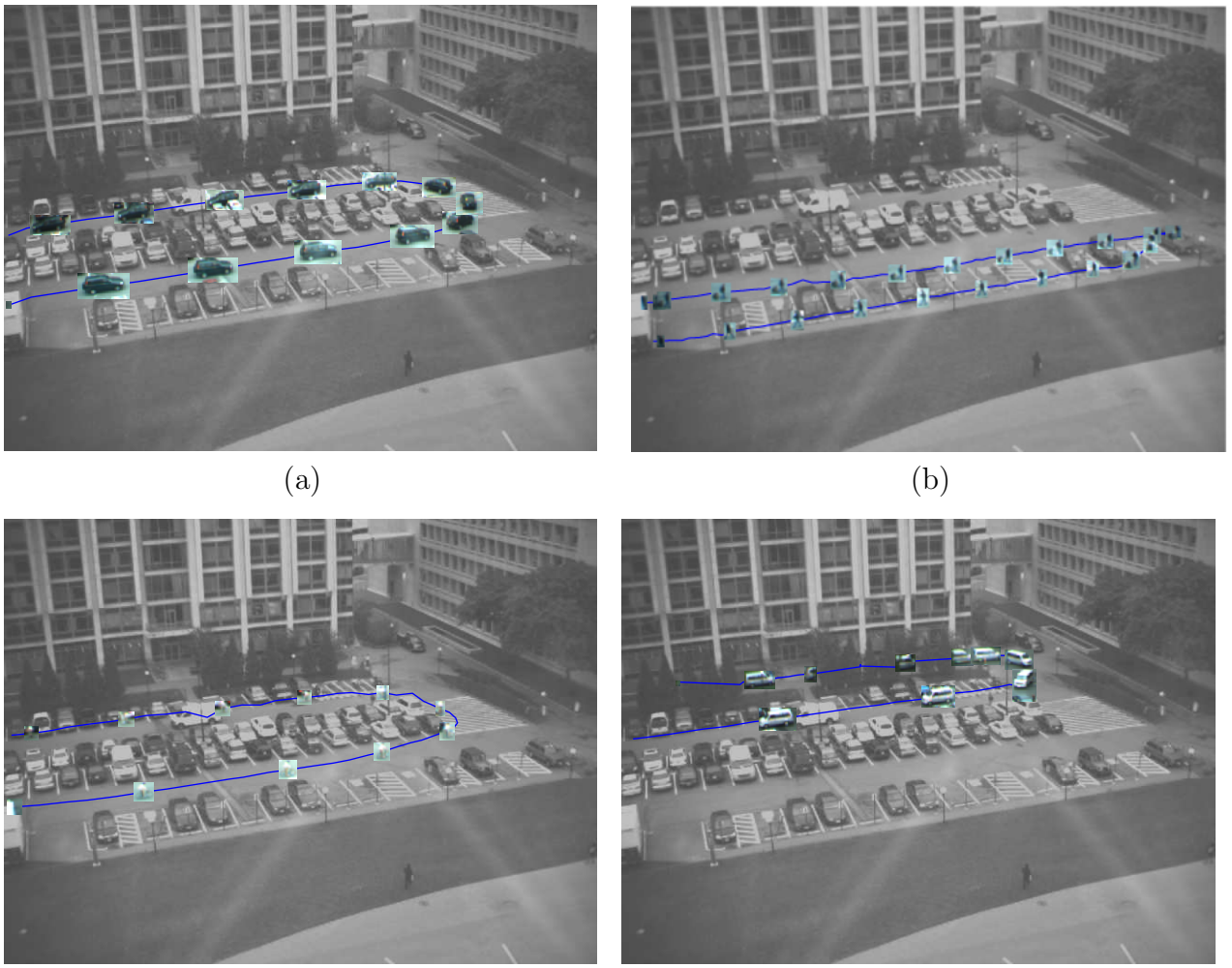
(a) (b)

Figure 1-3: (a) A sample of ca. 3000 object tracks (about 8% of our data set) shown as black trajectories. (b) The tracks from (a) clustered into 100 groups using our spectral clustering approach based on similarity along position, size, velocity and direction of motion. Color indicates cluster assignment (colors are reused).

model by first projecting them into the spectral embedding space and then calculating their likelihood under the mixture model.

Figure 1-4(b-c) shows examples of unusual activities detected using the track-based scene model. In (b), a person walked across the parking lot, dragging a large object, deposited the object on the grass at the far end of the lot and then walked back on the sidewalk. The unusual aspects of this activity are both the path that the person took through the scene as well as the change in size as he left the dragged object behind. Other examples include (c) a person riding a bicycle along a path that is common for cars but unlikely for objects of smaller size and (d) a car driving through a pedestrian zone and emerging into the parking lot. Note that in the first two examples, the individual observations of the person and the cyclist would not appear unusual, as people often walk or bike through the parking lot as well as along the pavement. Consequently, these tracks would not appear unusual under the observation-based scene model. Many unusual tracks would also not appear to be unlikely under a path-based model that did not consider additional attributes of the objects such as size, velocity and direction of motion.

Once an activity is flagged as unusual as compared to the learned model, it is possible



(a) (b) (c) (d)

Figure 1-4: (a) An example of typical activity: a car driving through the parking lot, searching for a space (with no success). (b)-(d) Examples of unlikely activities: (b) a person dragging an object across the parking lot, depositing it and walking back (both the path the person took and the size change after the object was left are unusual), (c) a person riding a bicycle along a path that is usual for cars but unusual for smaller objects, (d) a car emerging out of a pedestrian zone into the parking lot.

to direct a high resolution camera to capture more detailed images of the event. In the case of the person dragging an object, this may make it possible to identify the object, which cannot otherwise be discerned from the wide angle view of the scene. This underscores the importance of being able to classify tracks relatively quickly, while the associated activity is still in progress.

1.4 Methods

Having established the motivation for our approach, we can now describe our scene learning methods in more detail. Let us start from the input: a sequence of images of the observed scene collected over a long period of time (in our case, a week). First we extract out of the images the tracks of moving objects by segmenting each image into moving and stationary regions and solving for the correspondences between connected components of the moving regions in subsequent images. We describe these low-level methods in detail in Chapter 2.

Once we have extracted the moving object tracks from the input images, the images themselves are discarded: the tracks are essentially a compressed representation of the video input. The rest of the methods in this dissertation operate directly on tracks or the observations contained therein. We present two different types of scene modeling approaches, both of which take a collection of examples and learn from them a mixture model distribution using a clustering method.

In the observation-based scene modeling approach, the collection of examples is a set of observations that have occured in a particular area of the scene. In the track-based approach, examples are moving object tracks. We use a central clustering method (see Chapter 3) to fit a mixture model to a collection of examples. In the case of tracks, a central clustering method is not directly useful because tracks cannot easily be represented in a space in which Euclidean distances are meaningful. Thus, we use a spectral clustering approach, in which we first project all examples into a high-dimensional embedding space based on pairwise similarities between examples. In order to avoid having to calculate pairwise similarities between all pairs of examples in our huge dataset, we employ the Nyström method, which

allows us to estimate the embedding space from a much smaller sample of the data. Once all examples are projected into the embedding space, we use a central clustering method to group them. Finally, we represent the resulting groups as a mixture distribution. We give the background for our clustering methods in Chapter 3. In Chapter 4, we describe how these learning methods can be used to learn a scene model from collections of observations and in Chapter 5, we show how they can be applied to learning a scene model from collections of tracks.

In order to achieve a good clustering of tracks using a spectral method, the similarity measure used to compare examples is crucial. As we saw in the previous section, many types of interesting activities can only be detected if multiple attributes of moving objects are considered. We use a similarity measure under which two tracks appear similar only if they have similar trajectories and spatially corresponding observations along the tracks have similar attributes.

Having learned a scene model represented as a mixture distribution, the final step of our algorithm is evaluating the likelihood of new examples under the scene model. We define a statistical anomaly score that is proportional to the negative likelihood of examples under the model. The output of our algorithm is a ranking of the observed activities by their statistical anomaly scores. The ranking can be used to detect unusual activities (by thresholding on the anomaly scores) or to task a high resolution camera to the most unusual activity in the scene.

In chapter 6, we evaluate our results both qualitatively, by visualizing components of the track-based scene model; and quantitatively, by correlating the statistical anomaly scores with human perceptual judgments and assessing the performance of the scene model as a classifier for unusual activities. We present our conclusions in chapter 7.

1.5 Contributions

The key contributions of our work to the field of scene modeling and anomaly detection can be summarized as follows:

- Our track-based scene modeling approach is capable of using a very large dataset for learning a model of object motion through the scene. Thanks to our use of approximate spectral clustering with the Nyström approximation, we are able to utilize a volume of data an order of magnitude larger than would otherwise be practical. This is necessary in order to effectively model active, complex scenes where the number of observed objects per hour reaches several hundred or more. We demonstrate results on a collection of *ca.* 40,000 tracks representing a week of activity in an active outdoor scene. In comparison, the most comprehensive related scene modeling approaches have used *ca.* 1,500 tracks for learning.

- The distance measure we use to compare moving object tracks allows for combining multiple object attributes in a principled manner. This makes it possible to reason about not only the path the objects take through the scene but also various aspects of the observations along those paths, such as size, velocity, direction of motion, etc. The framework allows for further descriptors such as appearance to be incorporated into the distance measure. Further, when converting distances to similarities, we use a parameter-free approach that eliminates the need to set the parameters empirically–a tedious and often biased process.

- The model representation is efficient enough for us to reason about incoming objects online. This includes the ability to reason about partial object tracks and thus makes it possible to maintain a belief distribution of how well the developing track fits each of the clusters in our model. We can then detect sudden changes in the belief distribution to identify surprising moments.

- To our knowledge, this work is the first to evaluate the performance of a statistical anomaly detection approach by comparing its responses with human perceptual judgments. We show that the negative log likelihood of examples under our scene model is correlated with the degree to which humans perceive those examples as unusual.

1.6 Notation

Throughout this document, we will generally use the following notation:

- Vectors are written in boldface, e.g., $\boldsymbol{v}$.
- $\mathbf{1}$ denotes a vector of ones.
- Matrices are written as Roman capital letters, e.g., M.
- Scalars are written in italics, e.g., S.
- Superscripts generally denote the time index, e.g., x^t is the value of x at time t.
- Subscripts denote the membership index in a vector or an ordered set.
- δ_{ij} is the Kronecker delta:

$$\delta_{ij} = \begin{cases} 1 & \text{for } i = j \\ 0 & \text{for } i \neq j. \end{cases}$$

- Probabilities are written as $\Pr(X = x)$ whereas probability density functions are written in lowercase, e.g., $p(x)$.
- $\text{diag}(\boldsymbol{x})$ denotes a square diagonal matrix with the vector $\boldsymbol{x}$ on the diagonal.

Chapter 2

Scene and Activity Analysis Background

A key goal of any visual surveillance system is to automatically determine when an observed scene contains unusual or unexpected activity. In the past this task was performed by a human expert: someone familiar with the scene who was able to recognize when something out of the ordinary occurred. A typical surveillance site may have so many sensors in different locations that it is no longer feasible for a person to monitor all of them. Machine vision systems are needed to mine the collected data for potentially interesting activity. This has fostered a new area of machine vision research—often broadly referred to as surveillance—aimed at the statistical modeling of scenes and activities.

In this chapter, we review some of the background that forms the foundation of many of the methods used in surveillance, including those presented in this dissertation. We then give an overview of the related work in attentive monitoring and activity analysis systems. Towards the end of the chapter we focus on the research efforts in scene modeling. The last section contains a discussion of the published work most directly related to this dissertation.

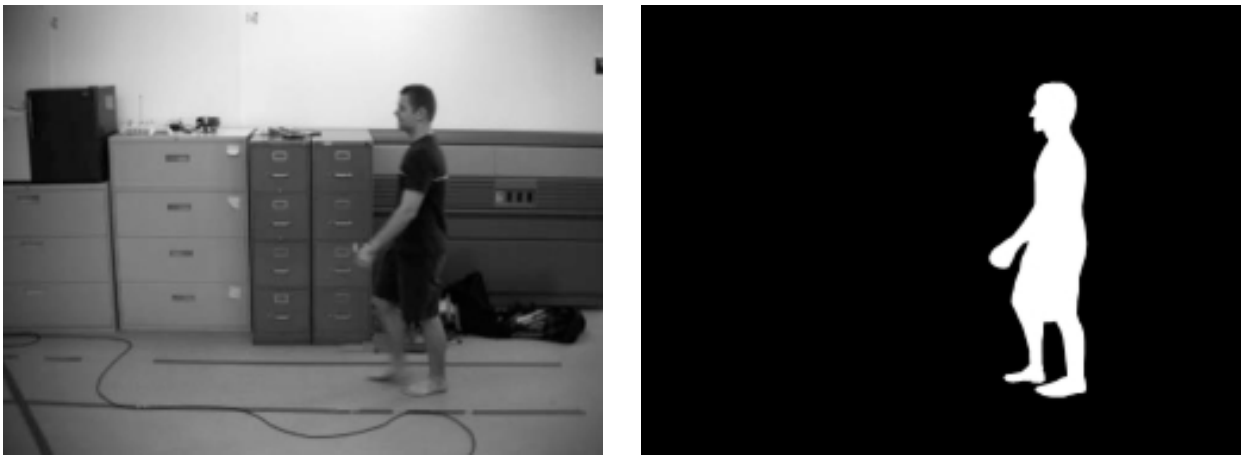


Figure 2-1: Illustration of motion segmentation. Left: input image. Right: segmentation into pixels corresponding to moving regions (white) and stationary regions (black). In this case the segmentation was performed manually.

2.1 Moving Object Segmentation

Among the many possible definitions of activity, one common theme stands out: motion. Without motion, it would be difficult to talk about activity. Detecting and quantifying motion in images is one of the foundations of most activity analysis research. In essence, given a new image of the scene, the problem in motion segmentation is to classify all of the pixels as either corresponding to moving or stationary parts of the scene. An example is shown in Figure 2-1. Naturally, some information about what the scene looked like just prior to acquiring the image is generally necessary to solve the problem. The amount of this prior information often determines the best approach.

Given only a few previous frames, optical flow may be the best choice. In optical flow, the focus is on finding correspondences between image patches in the current frame and corresponding patches in previous frames. This patch-wise registration is usually formulated as an optimization problem subject to some smoothness constraints. The solution yields a flow field, in which the vector at each image location indicates the corresponding location in the previous frame. We refer the reader to the seminal papers by Lucas and Kanade and by Horn and Schunck for an introduction to this topic [3, 15].

In the setting of monitoring systems with a stationary overview camera observing the

scene at all times, much more prior information about the scene is available for motion segmentation. A family of approaches termed background subtraction take advantage of this large amount of prior information. In background subtraction, the assumption is that if images of a particular scene location are collected over a long period of time, most of the collected frames will not contain motion and so it is possible to learn a pixel-level model of the appearance of the scene in the absence of any moving objects. A comparison of a new frame from the camera with the learned model then yields a segmentation, where each pixel is classified as foreground, i.e. belonging to a moving object, or background, i.e. corresponding to a part of the scene unoccluded by moving objects. Of course, one of the problems with this idea is that the appearance of the unmoving parts of the scene can (and does) change over time. In addition to lighting changes (for instance due to changing cloud cover or changing time of day), parts of the scene may become occluded by objects that have stopped moving, such as parked cars, or revealed when stationary objects begin moving.

2.1.1 Adaptive Background Subtraction

One of the most often used background subtraction methods, proposed by Stauffer and Grimson in 1999, overcomes this problem by using an adaptive approach, in which the background model is estimated online and changes in the appearance of the scene are gradually incorporated into the model [39]. Because this method is a critical component of our data collection, we describe it in detail here.

To understand the approach, we will consider a single pixel at the location (x, y) in the image. Let the grayscale value of that pixel in the frame at time t be I^t. This pixel value is proportional to the amount of light reflected from a particular location in the scene through the optics of the camera onto a photosensitive patch in the sensor. Thus it is determined by the properties of the corresponding scene location and the camera parameters. Because of measurement noise, variations in physical properties of the scene (such as dust in the air, lighting changes, etc.) and, of course, occlusion by any moving objects, the pixel value will not be the same in every frame. However, so long as most of the time the corresponding

scene location is unoccluded, we can reasonably assume that the pixel value is generated according to some probability distribution. We will also assume that each pixel value is picked independently of the other pixel values. The task now is to learn the parameters of the model M that generates values for the pixel at image coordinates (x, y).

Stauffer and Grimson do this by fitting a mixture of Gaussians to the history of pixel values seen so far. Let $l(t)$ indicate the index of the mixture component to which the pixel value observed at time t belongs. A pixel process model M with k components is then specified by the means ($\boldsymbol{\mu}$), variances ($\boldsymbol{\sigma}$) and priors ($\boldsymbol{\pi}$) of each mixture component:

$$
\begin{aligned}
t &\in \{1, .., T\}, && (2.1)\\
j &\in \{1, .., k\}, && (2.2)\\
M &= \{M_j\}, && (2.3)\\
M_j &= (\pi_j, \mu_j, \sigma_j), && (2.4)\\
\mu_j &= \frac{1}{\sum_{t:l(t)=j} 1} \sum_{t:l(t)=j} I^t, && (2.5)\\
\sigma_j^2 &= \frac{1}{\left(\sum_{t:l(t)=j} 1\right) - 1} \sum_{t:l(t)=j} (I^t - \mu_j)^2, && (2.6)\\
\pi_j &= \frac{\sum_{t:l(t)=j} 1}{T}. && (2.7)
\end{aligned}
$$

Given a new pixel value, we can calculate its likelihood under the model:

$$
\begin{aligned}
L(I^t) &= p(I^t|M) && (2.8)\\
&= \sum_j \pi_j p(I^t|M_j) && (2.9)\\
&= \sum_j \pi_j \frac{1}{\sigma_j\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma_j^2}\left(I^t - \mu_j\right)^2\right). && (2.10)
\end{aligned}
$$

Having learned a model of what the background usually looks like, background subtrac-

tion amounts to thresholding on the likelihood of the newly observed pixel value: whenever the likelihood exceeds some threshold, the pixel is classified as background because it matches the model well; otherwise it is classified as foreground.

To use the above approach in practice, we would have to keep in memory some number of previous frames and at each time step estimate the parameters of the Gaussian mixture model. To do this exactly using an expectation maximization [10] approach would be demanding both in terms of memory usage and computational complexity. Instead, the Stauffer and Grimson algorithm estimates the mixture model parameters using an online k-means algorithm. The basic idea is to determine whether a new observation is explained by one of the components of the mixture model. If so, the parameters of that mixture components are adjusted to take account of the new evidence. If not, a new mixture component is added or, alternatively, the least likely mixture component is replaced. The pixel is classified as background if it matches a salient mixture component—one for which enough consistent evidence has been observed.

The algorithm (summarized as Algorithm 2.1) work as follows. Suppose that at time $t-1$ the pixel process model at location (x, y) has k mixture components. Given a new pixel value I^t we take the following steps to classify the pixel and update the mixture parameters:

1. Sort the k mixture components according to their "saliency." Intuitively, a mixture component is more salient if it is supported by a larger amount of evidence (i.e. if its prior is high) and if the evidence is more consistent (i.e. if the variance is small). Accordingly, Stauffer and Grimson order the mixture components by the ratio π/σ. The first B components for which the total evidence is larger than some threshold T are considered part of the background model:

$$B = \underset{b}{\operatorname{argmin}} \sum_{i}^{b} \pi_i^t > T. \tag{2.11}$$

2. Calculate the likelihood $L_j(t)$, where $j \in \{1, ..., k\}$, of the new observation under each

Algorithm 2.1 Stauffer and Grimson background subtraction

Input: Current model $M_j^{t-1} = (\pi_j^{t-1}, \mu_j^{t-1}, \sigma_j^{t-1}), 1 \leq j \leq k$; new pixel value I^t; learning rate α; threshold γ.

Output: Classification $foreground(I^t)$; new model $M_j^t = (\pi_j^t, \mu_j^t, \sigma_j^t)$.

Sort M_j^{t-1} s.t. $M_m^{t-1} \leq M_n^{t-1} \iff \frac{\pi_m^{t-1}}{\sigma_m^{t-1}} \geq \frac{\pi_n^{t-1}}{\sigma_n^{t-1}}$

$B \leftarrow \min\{b : \sum_i^b \pi_i^t > T\}$

$match \leftarrow \text{argmax}_j \, p(I^t | M_j^t)$

if $||I^t - \mu_{match}|| > \gamma\sigma_{match}$ **then** // either replace last Gaussian

$match \leftarrow k$

$\pi_k^t \leftarrow \pi_0$

$\mu_k^t \leftarrow I^t$

$\sigma_k^t \leftarrow \sigma_0$

else // or update matched Gaussian

$\rho \leftarrow \alpha p(I^t | M_{match}^{t-1})$

$\mu_{match}^t \leftarrow (1-\rho)\mu_{match}^{t-1} + \rho I^t$

$(\sigma_{match}^t)^2 \leftarrow (1-\rho)(\sigma_{match}^{t-1})^2 + \rho(I^t - \mu_{match}^t)^T(I^t - \mu_{match}^t)$

for all j **do** // update priors

$\pi_j^t \leftarrow (1-\alpha)\pi_j^{t-1} + \alpha\delta_{j,match}$

if $match \leq B$ **then** // classify

$foreground(I^t) \leftarrow false$

else

$foreground(I^t) \leftarrow true$

mixture component:

$$L_j(t) = p(I^t|M_j). \tag{2.12}$$

3. Find the mixture component that best explains the new observation:

$$match = \underset{j}{\operatorname{argmax}}\, L_j(t). \tag{2.13}$$

4. If the best matching mixture component does not explain the new observation, i.e. if I^t is more than γ standard deviations from the mean, replace the mixture component with the lowest prior with a new Gaussian with an initial prior π_0, and an initial variance σ_0. Otherwise, update the parameters of the best matching mixture component:

   if $||I^t - \mu_{match}|| > \gamma\sigma_{match}$:

$$\begin{aligned} \pi_k^t &= \pi_0, \\ \mu_k^t &= I^t, \\ \sigma_k^t &= \sigma_0, \\ match &= k; \end{aligned} \tag{2.14}$$

   otherwise:

$$\begin{aligned} \mu_{match}^t &= (1-\rho)\mu_{match}^{t-1} + \rho I^t, \\ (\sigma_{match}^t)^2 &= (1-\rho)(\sigma_{match}^{t-1})^2 + \rho(I^t - \mu_{match}^t)^T(I^t - \mu_{match}^t), \end{aligned} \tag{2.15}$$

   where $\rho = \alpha p(I^t|M_{match}^{t-1})$ and α is a learning rate.

5. Adjust the mixture component priors. Intuitively, we want to increase the prior for the components for which we have seen more evidence and decrease the prior for the other components.

$$\pi_j^t = (1-\alpha)\pi_j^{t-1} + \alpha\delta_{j,match}, \tag{2.16}$$

Figure 2-2: Example adaptive background subtraction result. Left: input image. Middle: segmentation obtained using an adaptive background subtraction method. Right: manual segmentation shown for reference. The automatic segmentation shows the typical camouflaging errors in locations where the person's clothes closely resemble the appearance of the background.

where $\delta_{i,j}$ is the Kronecker delta, i.e. $\delta_{i,j} = 1$ if $i = j$ and $\delta_{i,j} = 0$ otherwise.

6. Classify the new pixel value as background if it is explained by one of the B (from step 1) salient mixture components of the background model:

$$foreground(I^t) = \begin{cases} \text{false} & \text{if } match \leq B; \\ \text{true} & \text{otherwise.} \end{cases} \tag{2.17}$$

Figure 2-2 shows an example background subtraction result compared with an ideal result obtained by manual segmentation. The result shows two of the typical problems that this method encounters. First, whenever the background is occluded by a moving object with a similar color appearance, those parts of the object that are similar to the background get incorrectly classified. This is often referred to as camouflaging. Another problem, related to the first, is that the silhouette often gets broken up into several segments. Various methods are used in practice to alleviate these problems. For instance, a connected component algorithm can be used to search the neighborhood of every foreground pixel and to associate foreground segments proximal to each other as parts of the same object. Additionally, morphological operations, such as dilation and erosion are often used to fill in "holes" in foreground silhouettes. In our data collection, we used the background subtraction method from [25], in which pixel processes in the image are modeled as nodes in a Markov

random field (MRF). The segmentation label (foreground or background) of each node in the model depends not only on the pixel value of that node, but also on the pixel values of neighboring nodes. Thus if the neighbors of a pixel are classified as foreground, the pixel itself is more likely to also be classified as foreground and vice versa. The method was shown to produce better silhouettes for object tracking than the original Stauffer and Grimson model for adaptive background subtraction.

2.2 Tracking

The output of background subtraction on a given input image is a set of blobs: connected foreground components that correspond to observations of moving objects in the image. The next step in obtaining tracks of moving objects from sequences of input images is solving for the correspondences between observations in subsequent images. This process is called tracking. One of the most popular ways of tracking moving blobs is with Kalman filters. For completeness we briefly describe our Kalman filter tracking algorithm in this section. For a more detailed treatment of the subject, we refer the reader to an introductory paper by Welch and Bishop [52].

Suppose that we are interested in estimating the state $\boldsymbol{s}$ of a moving object from observations consisting of the segmentation step described in the previous section. We will take the state to be the vector $\boldsymbol{s} = [x, y, \dot{x}, \dot{y}, a]$, where (x, y) are the image coordinates of the object's centroid, $(\dot{x}, \dot{y})$ is the instantaneous velocity of the centroid and a is the area of the bounding box. For simplicity, we will assume that the object's velocity and size (and thus the area of the bounding box) are constant and thus the state evolves according to the following linear dynamics:

$$\boldsymbol{s}^t = \mathrm{A}\boldsymbol{s}^{t-1} + w^{t-1}, \tag{2.18}$$

where based on our constant velocity and area assumptions, A is

$$\mathrm{A} = \begin{bmatrix} 1 & 0 & \triangle t & 0 & 0 \\ 0 & 1 & 0 & \triangle t & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.19}$$

The random variable w represents the process noise and is assumed to be normally distributed. Because the true state is not directly observable, the best we can do is predict it probabilistically from the observations and the process dynamics. The estimate at time $t-1$ is a Gaussian with mean $\hat{\boldsymbol{s}}^{t-1}$ and covariance matrix P^{t-1}. From this estimate, we can predict the *a priori* (i.e. prior to seeing a new observation) state $\hat{\boldsymbol{s}}_*^t$ at time t as:

$$\hat{\boldsymbol{s}}_*^t = \mathrm{A}\hat{\boldsymbol{x}}^{t-1}, \tag{2.20}$$

$$\mathrm{P}_*^t = \mathrm{A}\mathrm{P}^{t-1}\mathrm{A}^T. \tag{2.21}$$

In our tracking algorithm, we keep an estimated state for each tracked object. Given a new input frame, we first predict the expected state of the object in the current frame. Then we perform background subtraction and for each tracked object, we search a neighborhood around the predicted position with size proportional to the predicted variance. If we find a foreground blob that matches the predicted state (position, velocity and size) well, we assume that blob to be a new observation for that object. Given this new observation $\boldsymbol{o}^t$, we can now update the state to obtain the *a posteriori* estimate. As is often done in practice to decrease computational complexity, we assume that the covariance matrix P is diagonal. Then the update equations are:

$$\hat{\boldsymbol{s}}^t = \hat{\boldsymbol{s}}_*^t + \lambda(\boldsymbol{o}^t - \hat{\boldsymbol{s}}_*^t), \tag{2.22}$$

$$\mathrm{P}^t = (1-\lambda)\mathrm{P}_*^t + \lambda\mathrm{diag}((\boldsymbol{o} - \hat{\boldsymbol{s}}_*^t)^2). \tag{2.23}$$

The learning rate λ is usually set to be much smaller than 1. Intuitively, this moves the

estimated state towards the new observation. If the new observation matches the predicted state perfectly, the variance decreases. Conversely, if there is a large difference between the predicted state and the observation, the variance increases.

2.3 Attentive Monitoring Systems

In the introduction to this dissertation, we used attentive monitoring systems as the motivation for our work because of their connection to the way in which the human visual system addresses the tradeoff between coverage and resolution.

Focus-of-attention systems based on combinations of stationary and taskable cameras have been shown to perform well in various tracking and surveillance scenarios. Work related to such systems usually deals with at least one of the following problems: camera calibration [38, 35, 40], scheduling of taskable cameras to acquire high-resolution images [23, 58], and motion segmentation in the high-resolution view [22, 24].

All of the work mentioned either fails to address the problem of where to direct attention in the scene or uses resource scheduling combined with a rudimentary attention system (such as detecting only people) to task the movable cameras. The purpose of this dissertation is to approach the problem of where to direct attention in a new scene in a data-driven, unsupervised fashion by modeling the usual pattern of activity and reasoning about new activity in terms of its likelihood under the model.

2.4 Scene Modeling and Anomaly Detection

Modeling patterns of activity in moving scenes has been of great interest to the computer vision community for more than a decade. Recent years have seen a surge of published work in this area, thanks to the development of robust low-level vision algorithms that make it possible to acquire and analyze large amounts of data. Many different approaches to modeling scenes and detecting unusual activity have been proposed—each with its advantages and drawbacks. In this section, we attempt to characterize the space of related work and

describe where our own approach falls within it.

Because few modern machine vision algorithms operate directly on images, perhaps the most distinguishing feature of any scene modeling approach is its input. In this sense, the field can be divided into two main groups. In the first, sequences of observations (tracks) of moving objects are extracted from images of the scene and analysis is performed on some representation of these tracks. The second group of approaches uses observations of moving objects without solving for the correspondences between them.

2.4.1 Track-Based Scene Models

The intuition behind analyzing tracks of moving objects rather than individual observations is simple: a sequence of observations of the *same* object is more informative than a single observation, just as a video is more informative than a single image. Scene modeling approaches that use moving object tracks generally group the observed tracks according to some measure of similarity, and then reason in some way about the resulting clusters of tracks. Thus the main differences between these various approaches are the following:

- Representation of object tracks,
- Similarity measure used to compare examples,
- Grouping method used to divide examples into clusters,
- Representation of clusters, and
- Anomaly detection method.

Of course, these choices are not independent: the choice of similarity measure often depends on the track representation, just as the choice of clustering method is influenced by the similarity measure, and so on. Because the ultimate goal should be a useful scene model, let us begin by listing what we consider to be desirable properties of a scene model:

- The full range of activities in the scene should be represented. This can be accomplished either by using a large dataset for learning or by having an online method that improves the model asymptotically.

- It should be possible to compare objects with respect to a multitude of attributes rather than just position. Scalability with respect to the number of attributes is also desirable.

- Classifying new examples (and thus detecting anomalies) should be efficient.

- The number of parameters to be set empirically should be kept to a minimum.

We will begin with a discussion of approaches that use different activity representations from ours and conclude with those that are most related to our work.

In one of the first published activity pattern modeling approaches, Johnson and Hogg [20] track moving objects and extract for each object path a set of 4D position and velocity flow vectors of the form $\boldsymbol{f} = (x, y, dx, dy)$. To reduce the dimensionality of the space of examples, they use a neural network to learn a set of prototypes for these flow vectors. The neural network takes the 4D flow vectors at its input nodes and produces a pattern of activation on the output nodes, each of which corresponds to a prototype. Sequences of flow vectors are then modeled as sequences of activations of the output nodes and are fed as inputs into another neural network with a layer of "leaky" neurons (see also [34, 45]). The output of the leaky neurons depends on the current as well as the previous input, and so, once trained, the second neural network models the probability density function (pdf) over the sequential relationships between prototypes. The output of the second neural network is then again quantized in the same fashion as was done for the flow vectors. One of the drawbacks of this method is that the number of prototypes, and therefore the sizes of the neural networks, would grow with the dimensionality of the observations. Furthermore, though the possibility of detecting anomalies using this approach is discussed, no anomaly detection results are shown.

A similar, improved vector quantization approach was proposed by Stauffer and Grimson [42]. Instead of just using the positions and instantaneous velocities of observations, they learn a codebook of prototypes that include the size of the object. As in the Johnson and Hogg approach, activities are represented as sequences of prototypes. However, rather than modeling the sequential relationships of prototypes directly, the approach estimates a co-occurence matrix, in which each entry corresponds to the frequency of an object track generating both prototypes. A hierarchical model is then learned by estimating at each level two dissimilar prototype distributions that best explain the co-occurence matrix. The likelihood of a new sequence under a particular class in the model is then the product of likelihoods of all of the observations in the sequence under that class. As with the previous approach, it is not clear how well the prototype representation would scale if more object attributes were included in the model. Another drawback is that because the learning relies on the co-occurence matrix, it may be sensitive to the tracker grouping observations from unrelated objects into the same sequences and thus erroenously increasing co-occurence counts for unrelated prototypes. As the goal of the method is the classification of observations, anomaly detection performance is not evaluated.

Neither of the above approaches directly models sequences of observations of moving objects, making it difficult to reason about activities consisting of a likely distribution of observations in an unlikely order. To address this problem, several methods aimed at modeling object tracks as time series have been proposed.

Hu et al. [17, 16] and Fu et al. [14] take the average Euclidean distance between the image coordinates of the i-th observation in both tracks as the spatial dissimilarity between object tracks. While the Euclidean distance is efficient to compute and was shown to be relatively robust to noise [56], it requires tracks to contain equal numbers of observations sampled at similar time intervals. This was achieved by rejecting observations less than a predefined interval apart and linearly interpolating between observations [16] or assuming a constant velocity model and padding estimated observations onto the ends of shorter tracks [17, 14]). Additionally, for the average Euclidean distance between the i-th observation in

each track to make sense, the start and end points of tracks have to be aligned. In the kinds of scenes analyzed—traffic intersections—this was usually true because sources and sinks were at the edges of the image and velocities tended to be constant at the beginning and end of each track. However, the method would not generalize well to more complex scenes, such as the one modeled in our work, nor would it gracefully handle tracking errors that cause object tracks to be split into segments.

Several methods of comparing tracks that do not require tracks to be aligned have been proposed. Buzan et al. [5] use the Longest Common Subsequence [7] as a measure of dissimilarity between tracks. This allows tracks of different lengths and tracks in different spatial regions to be compared, but it is not clear how object attributes other than image coordinates of the observation could be incorporated. Additionally, the comparison measure takes several parameters that must be set by hand.

Porikli et al. [32, 31] fit a Hidden Markov Model (HMM) to each object track. As a reminder, in a (first-order) Markov Model, states correspond to observations, and the distribution over future states depends only on the present state. In a *Hidden* Markov Model, states are not directly observable (thus they are considered hidden) and instead generate observations according to some distribution over possible outputs. Porikli et al. [31] use a feed-forward HMM to model the first-order temporal relationships between the spatial locations of observations. They use a measure of mutual fitness, i.e. how well the model for one track explains another track, as the similarity between examples. In [32], they include comparison of other attributes such as size and velocity by comparing histograms of attributes accumulated over the entire track and show some promising anomaly detection results. Though histograms do capture the distribution of attributes along the track, they fail to capture the underlying temporal relationships. For instance, the track of a person jogging, slowing down to walk across the street and then resuming his jog may have the same velocity histogram as the track of a person walking, then running across the street and slowing down to a walk, even though the two would probably appear as quite different to an observer (as well as to our method).

In all of the above approaches, as well as in ours, the learning is a batch process. Online scene learning methods, in which a model is estimated on the fly, without holding the entire history of observations in memory, have also been recently suggested. Naftel and Khalid [27, 26] use the Discrete Fourier Transforms coefficient space to compare time series of x and y coordinates (no other object attributes are compared). They describe an online method that uses self-organizing maps to learn a set of patterns (classes) from the inputs. However, given the model, to determine if a new example is anomalous, they use the k nearest neighbors algorithm, which requires comparing the example to a set of classified examples. Thus while the scene modeling is online, the anomaly detection is not.

To compare object tracks in our work, we use a modified Hausdorff distance, which relates an observation in one track with the closest (in image coordinates) observation in the other tracks. Several other published approaches have used variants of this type of distance measure. Junejo et al. [21] cluster trajectories based on the Hausdorff distance between time series of image coordinates and then estimate a path envelope for each resulting cluster of paths. To classify an example, they evaluate how well it fits within the path envelope of the best matching cluster and, if the match is good, they further compare the new track's average velocity and curvature with the velocity and curvature distributions in the cluster. They detect anomalies by thresholding at each of the subsequent comparison steps. In addition to the need to set a number of thresholds by hand, this type of anomaly detection suffers from similar problems as the use of histograms: only the distributions of various attributes are modeled rather than the temporal relationships between observations.

The approach that is most related to our work was proposed by Wang et al. [47]. They use a modified Hausdorff distance, in which the average difference between the closest obserations along two tracks is considered to be the dissimilarity between the tracks. The difference between observations is taken to be the sum of the Euclidean distance between the image coordinates and a weighted sum of the differences between various other attributes, such as size and velocity. This has the advantage of directly comparing time series of not only coordinates but also other attributes, but the drawback is the need to set the weights

for different attributes by hand. In contrast, our approach combines different attributes in a principled manner, based on the local variances in those attributes, so that each attribute contributes equally to the final dissimilarity. Another major difference between this approach and our work is the representation used for the scene model when classifying new examples. Wang et al. represent classes as distributions over the attributes of the examples belonging to each class and distributions of sources and sinks in the scene. A new example is considered anomalous if it contains observations that are unlikely under these distributions. In contrast, we represent the scene model as a mixture of Gaussians in the spectral embedding space and classify new examples based on their likelihood under the mixture model, thus evaluating each track as a whole, rather than as a collection of examples.

Thus far, all of the methods we have discussed in this section were based on the history of observations of moving objects in the scene. However, it is also possible to detect anomalies based on *a priori* models of agent behavior. For example, Dee and Hogg [8, 9] classify tracks as anomalous based on how well the object behavior can be explained as navigating towards a goal around a set of obstacles. This has obvious disadvantages, such as the need to specify (or learn) a set of goals and obstacles and the inability to incorporate observed evidence into the model. However, the anomaly detection method performs well when compared with human judgments on a set of examples. As we will show in Chapter 6, our anomaly detection method also correlates well with human judgments. This is significant because to our knowledge, no other statistical anomaly detection method has been evaluated in this fashion. In fact, in all of the statistical scene modeling approaches discussed above, the only performance analysis of the anomaly detection framework (if any), consisted of a few examples of detected activity.

The various similarities and differences between the above approaches are summarized in table 2.1. A final aspect of the related work that we have thus far not discussed is the size of the dataset used for clustering and analysis. As can be seen from table 2.1, our dataset was at least an order of magnitude larger than those used in the related work, due to the fact that we performed our experiment on a much more complex scene using observations

collected over a full week of tracking.

2.4.2 Low Level Activity Analysis

Track-based models of activity are only useful in scenes where it is possible to solve for the correspondences between observations of moving objects. Examples of scenes where tracking objects would be very difficult are scenes captured by moving cameras or very crowded or cluttered scenes, such as subway stations. In the absence of the correspondences between observations, it is still possible to reason about activity in the scene based on low-level visual features.

Wang et al. [46] apply techniques from the field of document analysis to the problem of modeling activity in a scene based on low-level motion features. They define their vocabulary as a codebook of local optical flow features, in which each word corresponds to optical flow with a particular direction and location in the image. The documents—short video sequences—are then represented as a bags of words. To learn the pattern of activity in the scene, Wang et al. fit a hierarchical Dirichlet process (HDP)[43] to an input set of documents, thus essentially learning a distribution over a set of "topics," each corresponding to a particular type of activity. They show promising results on a traffic intersection scene, where they are able to detect interesting events such as jay-walking. Their work is related to other approaches inspired by document analysis, in which global image features are used, such as spatial histograms of optical flow magnitude [57] or temporal pyramids [54].

2.5 Summary

In this chapter we discussed how observations (or sequences of observations) of moving objects can be obtained from video input, and we placed our work in the context of other scene modeling and anomalous activity detection approaches. In the next chapter, we give an overview of the unsupervised learning methods needed for our observation-based and track-based scene modeling approaches.

Approach	Features	Similarity Measure	Clustering Method	Model Representation	Dataset Size	Anomaly Detection Evaluation
Hu et al. [16]	time series (position, velocity, size)	Euclidean	multistage fuzzy k-means	Gaussian chains	1,200	qualitative
Porikli [31]	HMMs (position, velocity, size)	mutual fitness	spectral	clusters	< 200	qualitative
Junejo et al. [21]	time series (position and curvature) + avg. velocity	Hausdorff	min-cut	average path + envelope	< 100	qualitative
Wang et al. [47]	time series (position, velocity, size)	modified Hausdorff	spectral	semantic regions + feature distributions	600	qualitative
Naftel & Khalid [27]	DFT of coordinate time series	Euclidean	self-organizing map	cluster centers	800	qualitative
Dee & Hogg [9]	coordinate time series	angular disparity	n/a	predicted paths to goals	300	quantitative
Ours	time series (position, velocity, size, time of day, ...)	modified Hausdorff	approximate spectral	Gaussians in embedding space	40,000	quantitative

Table 2.1: Comparison of trajectory-based approaches for detecting unusual events.

Chapter 3

Overview of Clustering Methods

In unsupervised learning, a common method of fitting a model to a set of examples is by clustering—grouping similar examples into clusters. The goal of clustering is to arrive at a set of classes such that the members of each class are similar to each other but dissimilar to members of other classes. This requirement is commonly expressed as a cost function, allowing the clustering algorithm to be formulated as an optimization problem. In this section, we describe two popular families of clustering algorithms: expectation maximization (along with k-means—a special case) and spectral grouping. Implementations of both types of clustering are used in our observation-based and track-based modeling algorithms.

3.1 Expectation-Maximization

In its most general form, the expectation-maximization (EM) method is a way to estimate the parameters of a probability distribution from incomplete data.[1]. Let us assume that we have a set of samples $D = \{\boldsymbol{d}_1, ..., \boldsymbol{d}_n\}$ taken independently from the same distribution. We want to find the parameters $\boldsymbol{\Theta}$ describing this distribution. However, to complicate matters, instead of observing the full set D, we can only see a corrupted data set with some of the features missing. We can think of this as the data set containing some good features and

[1]For a good introduction to EM and its application to clustering, see [4] or [11]. Much of the material in this section is based on these two sources.

some bad (missing) ones. Grouping all the good features into a separate set D_g and the missing ones into the set D_b, we have $D = D_b \bigcup D_g$. We can then define the following function, often referred to as the central equation of expectation maximization [4]:

$$Q(\boldsymbol{\Theta}^t, \boldsymbol{\Theta}^{t-1}) = \mathcal{E}_{D_b}[\ln p(D_g, D_b|\boldsymbol{\Theta}^t)|D_g, \boldsymbol{\Theta}^{t-1}]. \tag{3.1}$$

We can think of this function as a way of taking the estimated parameters $\boldsymbol{\Theta}^{t-1}$ at time $t-1$, and using them to evaluate a new, improved estimate $\boldsymbol{\Theta}^t$ at time t. It therefore makes sense that the current estimate $\boldsymbol{\Theta}^{t-1}$ be treated as a constant in the above expression. Similarly, the "good" data features D_g are treated as constant as well, whereas the bad feature vector, D_b, is a random variable. We can then interpret the right side of equation (3.1) as evaluating the expected log-likelihood of the new parameter vector given the full data set, but marginalized with respect to the previous best estimate and thus evaluating the two against each other. This evaluation is called the E-step of the expectation-maximization algorithm.

Having developed a measure to compare new parameter estimates to previous ones, we now need to select a new estimate which maximizes the function Q as defined above. We can express this relationship formally:

$$\boldsymbol{\Theta}^t = \underset{\boldsymbol{\Theta}}{\operatorname{argmax}}\, Q(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{\boldsymbol{t-1}}). \tag{3.2}$$

This optimization is referred to as the M-step of the EM algorithm. After an initial estimate for the parameter vector $\boldsymbol{\Theta}$ is chosen, the E-step and M-step are alternated to iteratively improve the estimate. One of the key properties of the EM algorithm is the fact that it guarantees the log-likelihood of the estimated parameter vector, given the data, to monotonically increase [10].

A concrete implementation would be hopeless unless we put some assumptions on the probability density of the entire data set. In practice, it is often assumed that the data have a distribution that can be approximated by a mixture of Gaussians. This leads to an elegant implementation of the algorithm, as we show in the next section.

3.1.1 EM for Mixtures of Gaussians

The EM algorithms lends itself readily to be applied to the problem of clustering if we think of each element of the full data set D as the pair $(\boldsymbol{x}, l)$, where $\boldsymbol{x}$ is an example observation and l is is the label of the cluster to which the observation belongs. In effect, the observations are our good features and the labels are our missing data. We can now assume that the missing labels form a vector $\boldsymbol{l}$ of independent random components governed by the probability distribution $p(\boldsymbol{l}|X, \boldsymbol{\Theta}) = \prod_{i=1}^{n} p(l_i|\boldsymbol{x}_i, \boldsymbol{\Theta})$, where $X = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$. Further, we will assume that the data was sampled from a multivariate Gaussian mixture distribution. Having made these assumptions, equation (3.1) becomes:

$$\begin{aligned} Q(\boldsymbol{\Theta^t}, \boldsymbol{\Theta^{t-1}}) &= \sum_{\boldsymbol{l} \in \Lambda} \ln p(X, \boldsymbol{l}|\boldsymbol{\Theta}^t) p(\boldsymbol{l}|X, \boldsymbol{\Theta}^{t-1}) \\ &= \sum_{\boldsymbol{l} \in \Lambda} \ln \prod_{i=1}^{n} P^t(l_i) p(\boldsymbol{x}_i|l_i, \boldsymbol{\theta}_i^t) \prod_{i=1}^{n} p(l_i|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1}) \\ &= \sum_{\boldsymbol{l} \in \Lambda} \sum_{i=1}^{n} \ln P^t(l_i) p(\boldsymbol{x}_i|l_i, \boldsymbol{\theta}_i^t) \prod_{i=1}^{n} p(l_i|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1}). \end{aligned} \tag{3.3}$$

In the above equation, $P^t(l_j)$ is the prior probability of the i-th mixture component at iteration t, $p(\boldsymbol{x}_i|l_i, \boldsymbol{\theta}_i^t)$ is the density of the i-th component with the parameter vector $\boldsymbol{\theta}_i^t$ and Λ is the space of all possible label assignments.

After some algebraic manipulation, we can express equation 3.3 as a sum of the following two terms:

$$Q(\boldsymbol{\Theta^t}, \boldsymbol{\Theta^{t-1}}) = \sum_{j=1}^{N} \sum_{i=1}^{n} \ln(P^t(j) p(j|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1})) + \sum_{j=1}^{N} \sum_{i=1}^{n} \ln(p(\boldsymbol{x}_i|j, \boldsymbol{\theta}_i^t) p(j|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1}). \tag{3.4}$$

Since we are assuming that the N mixture components have a multivariate normal density, the term $p(\boldsymbol{x}_i|j, \boldsymbol{\theta}_i^t)$ from the above equation expands to the following:

$$p(\boldsymbol{x}_i|j,\boldsymbol{\theta}_i^t) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}_j^t|^{1/2}} \exp[-\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{\mu}_j^t)^T(\boldsymbol{\Sigma}_j^t)^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_j^t)]. \tag{3.5}$$

Having thus simplified the E-step, we are finally ready for the M-step of the algorithm. We want to minimize the function Q from (3.4) with respect to the new parameter estimate $\boldsymbol{\Theta}^t$. This parameter vector consists of the following components:

$$\boldsymbol{\Theta}^t = \bigcup_{j=1}^{N}\{(P^t(j), \boldsymbol{\Sigma}_j^t, \boldsymbol{\mu}_j^t)\}, \tag{3.6}$$

where $P^t(j)$, $\boldsymbol{\Sigma}_j^t$ and $\boldsymbol{\mu}_j^t$ are, respectively, the new estimates of the prior, covariance matrix and mean of the j-th mixture component.

We can approach this optimization problem by minimizing each of the two terms on the right side of equation 3.4 separately. The first term yields the solution for $P^t(j)$ after setting its partial derivative with respect to $P^t(j)$ to 0 and adding the normalizing constraint $\sum_{i=1}^{N} P^t(j) = 1$. The resulting update equation for $P^t(j)$ is as follows:

$$P^t(j) = \frac{1}{N}\sum_{i=1}^{n} p(j|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1}). \tag{3.7}$$

Differentiating the second term of equation 3.4 with respect to $\boldsymbol{\Sigma}_j^t$ and $\boldsymbol{\mu}_j^t$ yields the remaining two update equations:

$$\boldsymbol{\mu}_j^t = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i p(j|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1})}{\sum_{i=1}^{n} p(j|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1})}, \tag{3.8}$$

$$\boldsymbol{\Sigma}_j^t = \frac{\sum_{i=1}^{n} p(j|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1})(\boldsymbol{x}_i - \boldsymbol{\mu}_j^t)(\boldsymbol{x}_i - \boldsymbol{\mu}_j^t)^T}{\sum_{i=1}^{n} p(j|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1})}. \tag{3.9}$$

These three update equations express simultaneously the E-step (evaluation of the Q function) and M-step (maximizing the Q function with respect to the new parameters) of the EM algorithm. They make implementation of the algorithm relatively straightforward: generate an initial guess for the parameters $\boldsymbol{\Theta}^0$ and then repeatedly update the parameter values until some threshold condition is met. Perhaps the only tricky part is evaluating the

term $p(j|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1})$ from equations 3.7-3.9. However, a simple application of Bayes' rule gives us:

$$p(j|\boldsymbol{x}_i, \boldsymbol{\Theta}^{t-1}) = \frac{p(\boldsymbol{x}_i|j, \boldsymbol{\Theta}^{t-1})P^{t-1}(j)}{\sum_{j=1}^{N} p(\boldsymbol{x}_i|j, \boldsymbol{\Theta}^{t-1})P^{t-1}(j)}$$

where $P^{t-1}(j)$ is the previous prior and $p(\boldsymbol{x}_i|j, \boldsymbol{\Theta}^{t-1})$ the previous density for the j-th mixture component.

3.2 K-Means

Because of the computational complexity of the EM algorithm, a simpler, faster clustering method is often preferable. One of most popular fast central clustering algorithms is k-means. It is really a special case of expectation maximization made faster by some simplifying assumptions. First, the Gaussian clusters are all assumed to have the same diagonal covariance matrix, i.e. the clusters are assumed to be spherical. Additionally, the prior over the clusters is assumed to be uniform. Finally, instead of a "soft" assignment of examples to clusters based on the likelihood, a hard assignment is used, in which an example belongs to the cluster with the closest (in terms of Euclidean distance) mean. Thus the k-means algorithm seeks to minimize the sum of squared Euclidean distances between the examples and their closest cluster means. The algorithm (see Algorithm 3.1) is popular because each iteration is $O(kNd)$, where k is the number of clusters, N is the number of examples and d is the dimensionality of the feature vectors. However, though the algorithm usually converges fast in practice, it has been shown that on certain (very specific) clustering problems, the algorithm is superpolynomial ($2^{\Omega(\sqrt{n})}$) [1].

We should note that because both EM and k-means are sensitive to the initialization, in practice they are frequently run multiple times for various initializations and the best clustering is chosen as the final solution. Further, k-means is often used to initialize the EM algorithm [11].

Algorithm 3.1 K-means clustering

Input: Set of vectors $X = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\}$; number of clusters k; threshold ϵ; maximum number of iterations $maxt$.

Output: Cluster means M; assignment vector $\boldsymbol{l}$ s.t. l_i is the label for $\boldsymbol{x}_i$.

```
Set M^0 = {μ_1^0, ...μ_k^0} to be a random sample from X
t ← 0
repeat
    t ← t + 1
    for i = 1 to N do                          // assign points to closest mean
        l_i = argmin_{j∈{1,...,k}} ||x_i − μ_j||
    for j = 1 to k do                          // reestimate means
        μ_j^t = 1/(Σ_{i:l_i=k} 1) Σ_{i:l_i=k} x_i
    M^t = {μ_1^t, ..., μ_k^t}
until Σ_{j=1}^k ||μ_j^{t−1} − μ_j^t|| < ε or t ≥ maxt
M = M^t
```

3.3 Spectral Grouping

Spectral clustering is a popular alternative to central clustering methods such as Gaussian expectation-maximization or k-means, in which the assumption is that the feature vectors in each class are centrally distributed around some prototype and that the Euclidean distance is a meaningful way of comparing examples. Spectral clustering methods take their name from spectral theory, which is concerned with the eigenvectors and eigenvalues of matrices, and spectral graph theory [6], in which properties of graphs are examined in terms of certain characteristic matrices. We will give an introduction to spectral clustering by describing a specific graph-theoretic variant called normalized cuts, first proposed by Shi and Malik [36, 37].

3.3.1 Normalized Cuts

Let us consider a set of N feature vectors (examples) $X = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\}$. If we think of each feature vector as a node in a complete weighted graph G, in which the edge weights correspond to the similarities between the adjacent examples, then we can solve the clustering problem by making cuts in the graph in such a way that similar examples remain

connected, while dissimilar groups of examples become disconnected. The problem is to arrive at a graph cut, i.e. a set of cuts made in the graph, that maximizes some "goodness" criterion. Intuitively, cutting edges with low weights is desirable because it separates dissimilar examples. Conversely, cutting edges with large weights should be penalized because it disassociates similar examples.

Suppose for simplicity that we want to partition our examples into two dissimilar disjoint groups A and B. If the weight of the edge between example $\boldsymbol{u}$ and $\boldsymbol{v}$ is $w(\boldsymbol{u}, \boldsymbol{v})$, then the cost of the cut can be quantified by summing the weights of the edges that are separated:

$$cut(A, B) = \sum_{\boldsymbol{u} \in A, \boldsymbol{v} \in B} w(\boldsymbol{u}, \boldsymbol{v}). \tag{3.10}$$

Minimizing the cost of the cut, i.e. finding the minimum cut, which is a well studied problem in graph theory, has been used with promising results in image segmentation [53]. However, one of the problems of minimum cut clustering is that it favors separating out small groups of isolated examples. To solve this problem, Shi and Malik suggest expressing the cost of the cut as a fraction of the total weight of edges emanating out of the nodes in either partition [37]. Let us consider partitioning the examples into two disjoint sets, A and B, as above. Then the total weight of edges coming from nodes in A is $assoc(A, X) = \sum_{\boldsymbol{u} \in A, \boldsymbol{v} \in X} w(\boldsymbol{u}, \boldsymbol{v})$. The normalized cut objective function can then be written as:

$$ncut(A, B) = \frac{cut(A, B)}{assoc(A, X)} + \frac{cut(A, B)}{assoc(B, X)}. \tag{3.11}$$

Intuitively, the cut that will minimize this criterion will partition the nodes such that the total weight of the cut will be small in proportion to the "connectedness" of the two partitions.

Another way to represent a cut is by defining an indicator vector $\boldsymbol{z}$ such that

$$z_i = \begin{cases} -1 & \text{for } \boldsymbol{x}_i \in A; \\ 1 & \text{for } \boldsymbol{x}_i \in B. \end{cases} \tag{3.12}$$

Let d_i be the degree of the i-th node (the total weight of edges adjacent to it) and let a_{ij} be the weight (affinity) of the edge between the i-th and j-th node. Then the cost of partitioning the graph according to the indicator vector $\boldsymbol{z}$ is

$$cut(\boldsymbol{z}) = \frac{\sum_{z_i<0, z_j>0} a_{ij}}{\sum_{z_i<0} d_i} + \frac{\sum_{z_i>0, z_j<0} a_{ij}}{\sum_{z_i>0} d_i}. \tag{3.13}$$

Shi and Malik show that minimizing the above cost is NP-complete. However, if the the indicator vector is allowed to take on real values, they show that the solution to the discrete problem can be approximated by thresholding on the second smallest eigenvector of the following generalized eigenvalue problem:

$$(\mathrm{D} - \mathrm{A})\boldsymbol{y} = \lambda \mathrm{D}\boldsymbol{y}. \tag{3.14}$$

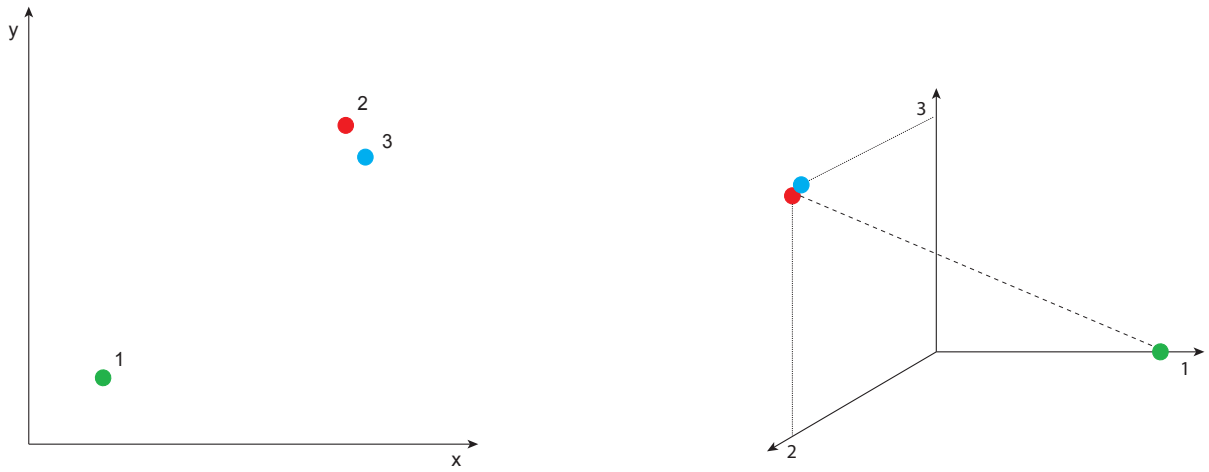
In the above eigenvalue system, $\mathrm{D} = \mathrm{diag}(\boldsymbol{d})$ is the diagonal degree matrix of the graph, in which the diagonal element d_i is the degree of the i-th node and $\mathrm{A} = (a_{ij})$ where $a_{ij} = w(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is called the affinity matrix of the graph. The matrix $\mathrm{L} = (\mathrm{D} - \mathrm{A})$ is called the Laplacian of the graph G. We omit the proof here (it can be found in [37]), but we will attempt to give some insight into why this seemingly fortuitous result makes intuitive sense.

First, let us consider the above generalized eigenvalue system more closely. We can rewrite it as a standard eigenvalue problem as follows:

$$\mathrm{D}^{-1/2}\mathrm{L}\mathrm{D}^{-1/2}\mathrm{D}^{1/2}y = \lambda \mathrm{D}^{1/2}y. \tag{3.15}$$

We can see from (3.14) that $\mathbf{1}$ is a generalized eigenvector, with eigenvalue 0: $(\mathrm{D}-\mathrm{A})\mathbf{1} = \mathrm{D}\mathbf{1} - \mathrm{A}\mathbf{1} = \boldsymbol{d} - \boldsymbol{d} = \mathbf{0}$. Furthermore, because the Laplacian matrix of a graph is positive semi-definite, the generalized eigenvalues will be non-negative and thus $\mathbf{1}$ is the smallest eigenvector. This makes sense, as the indicator vector $\mathbf{1}$ corresponds to the degenerate cut with cost 0 that partitions X into X and $\emptyset$.

Let us consider a generalized eigenvector $\boldsymbol{v}$ with eigenvalue λ. With some algebraic manipulation, we can show that $\mathrm{D}^{1/2}\boldsymbol{v}$ is an eigenvector of the normalized affinity matrix



(a) (b)

Figure 3-1: Simple illustration of the meaning of the affinity matrix eigenvectors: (a) simple dataset consisting of three 2D points; (b) the points in the similarity space spanned by the 3D columns of the affinity matrix. The dashed line shows the principal axis, i.e. the direction of the largest eigenvector.

$\mathrm{A}' = \mathrm{D}^{-1/2}\mathrm{A}\mathrm{D}^{-1/2}$ with eigenvalue $1 - \lambda$ (note that $\mathrm{A}' = (a_{ij}/\sqrt{d_i d_j})$ is just the affinity matrix A normalized by the product of the square roots of the row sum and column sum) [48]:

$$
\begin{aligned}
(\mathrm{D} - \mathrm{A})\boldsymbol{v} &= \lambda \mathrm{D}\boldsymbol{v} && (3.16)\\
\mathrm{A}\boldsymbol{v} &= (1-\lambda)\mathrm{D}\boldsymbol{v} && (3.17)\\
\mathrm{A}\mathrm{D}^{-1/2}\mathrm{D}^{1/2}\boldsymbol{v} &= (1-\lambda)\mathrm{D}^{1/2}\mathrm{D}^{1/2}\boldsymbol{v} && (3.18)\\
\mathrm{D}^{-1/2}\mathrm{A}\mathrm{D}^{-1/2}\mathrm{D}^{1/2}\boldsymbol{v} &= (1-\lambda)\mathrm{D}^{1/2}\boldsymbol{v} && (3.19)\\
\mathrm{A}'\mathrm{D}^{1/2}\boldsymbol{v} &= (1-\lambda)\mathrm{D}^{1/2}\boldsymbol{v}. && (3.20)
\end{aligned}
$$

Because the smallest generalized eigenvector is $\mathbf{1}$, it follows that the largest eigenvector (with eigenvalue 1) of A' is $\mathrm{D}^{1/2}\mathbf{1}$. Thus the second smallest generalized eigenvector is actually a component-wise ratio of the two *largest* eigenvectors of A'.

Why should the largest eigenvectors of the normalized affinity matrix be helpful in our

clustering problem? What is the intuitive meaning of these eigenvectors? Let us consider the very simple case with only three data points shown in Figure 3-1(a). Suppose that points 2 and 3 are maximally similar and point 1 is maximally dissimilar from the other two points. The affinity matrix will be:

$$\mathrm{A} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

We can think of the i-th row of A as a representation of the i-th data point in a three-dimensional similarity space, in which each axis corresponds to one of the three points. As we can see from Figure 3-1(b), the points are well separated into two clusters by the principal axis (i.e. along the largest eigenvector).

So far we have only discussed how to use the eigenvectors of the normalized affinity matrix to separate the data into two clusters. To group the examples into k-clusters, the bipartitioning can be performed recursively [37] or, alternatively, a subset of the largest eigenvectors can be used to separate the data into k clusters directly [13, 28]. For an approximately block-diagonal matrix with k blocks, the k largest eigenvectors will contain clustering information [48]. However, in practice it has been observed that using smaller subsets of these eigenvectors for clustering makes the algorithm more robust [13, 29].

In our work, we use the spectral clustering algorithm from [13] to cluster moving object tracks. In the algorithm (summarized as Algorithm 3.3.1), the first $N_E \ll N$ largest eigenvectors of the N x N normalized affinity matrix are computed, scaled by the square roots of the row sums as in equation (3.20), and stacked as column vectors to form the embedding matrix U_E. Note that the first scaled eigenvector can be omitted because it is constant. The rows of the embedding matrix are then grouped into k clusters using a central grouping method such as k-means.

Each row in the embedding matrix U_E can be thought of as the projection of the corresponding example (as represented by a row of the affinity matrix) into the spectral embedding space spanned by the first N_E largest eigenvectors of the normalized affinity matrix. We will

use this fact in Chapter 5, where we represent each cluster as a Gaussian in the spectral embedding space.

Algorithm 3.2 k-way Normalized Cuts spectral clustering

Input: N x N Similarity matrix A $= (a_{ij})$ where $0 \leq a_{ij} \leq 1$ is the similarity between points i and j ; number of clusters k; dimension of embedding space $N_E \ll N$.

Output: Assignment vector $\boldsymbol{l}$ s.t. l_i is the label for i-th datapoint.

Calculate degree matrix D $= (d_{ij})$ where $d_{ij} = \delta_{ij} \sum_{n=1}^{N} a_{jn}$

Scale A: A$' = \mathrm{D}^{-1/2}\mathrm{A}\mathrm{D}^{-1/2}$

Diagonalize A$'$: A$' = \mathrm{U}\Lambda\mathrm{U}^T$

Scale U: U$' = \mathrm{D}^{-1/2}\mathrm{U}$

$\mathrm{U}_E \leftarrow$ columns 2 through $N_E + 1$ of U$'$

Cluster rows of U_E using k-means (Algorithm 3.1)

Set $\boldsymbol{l}$ to the output of k-means

3.3.2 Nyström Approximation

One of the limitations of spectral grouping methods is that computing the affinity matrix for a dataset of size N requires $O(N^2)$ pairwise affinity computations. In many modern clustering problems, large data sets are not rare. For instance, a week of tracking data from the moving scene analyzed in our work contains roughly 40,000 moving object tracks. Fortunately, it is possible to estimate the eigenvectors of the full affinity matrix from a randomly sampled subset of its rows using the Nyström method [2, 33].

The Nyström method is essentially a way to approximately solve integral equation eigenfunction problems. Let us consider the eigenfunction problem

$$\int_0^1 w(x,y)\phi(y)dy = \lambda\phi(x), y \in [0,1]. \tag{3.21}$$

Solutions to this problem are pairs of eigenvalue λ and eigenfunction ϕ. Using a simple quadrature rule, we can approximate the integral on the left side of the equation by summing over a set of quadrature points $s_j, j = 1, ..., n$. This leads to the following equation, the solutions of which are pairs of approximate eigenvalues $\hat{\lambda}$ and approximate eigenfunctions $\hat{\phi}$:

$$\frac{1}{n}\sum_{j=1}^{n} w(x, s_j)\phi(s_j) = \hat{\lambda}\hat{\phi}(x). \tag{3.22}$$

The trick of the Nyström method is to now set $x = s_i$, which leads to the following set of equations:

$$\frac{1}{n}\sum_{j=1}^{n} w(s_i, s_j)\phi(s_j) = \hat{\lambda}\hat{\phi}(s_i). \tag{3.23}$$

Let $\hat{\lambda}_m$ and $\hat{\phi}_m$ be the m-th eigenvalue and eigenfunction. By substituting from (3.23) into (3.22), we can extend $\hat{\phi}_m$ from the set of Nyström points s_i to an arbitrary point x:

$$\hat{\phi}_m(x) = \frac{1}{n\hat{\lambda}_m}\sum_{j=1}^{n} w(x, s_j)\hat{\phi}_m(s_j). \tag{3.24}$$

Note that if we set $W = (w_{ij})$ where $w_{ij} = w(s_i, s_j)$ we can write the system of equations in (3.23) in matrix form:

$$\mathrm{W}\hat{\Phi} = n\hat{\Phi}\Lambda, \tag{3.25}$$

where the m-th column of $\hat{\Phi}$ is the eigenfunction $\hat{\phi}_m$ evaluated at the Nyström points and Λ is a diagonal matrix with the eigenvalues $\hat{\lambda}_m$. Similarly, we can express the Nyström extension (3.24) of the eigenfunctions to a new set of points x_i in matrix form. Let $\bar{\mathrm{W}} = (\bar{w}_{ij})$ where $\bar{w}_{ij} = w(x_i, s_j)$ and $\bar{\Phi}$ be a matrix in which the element at (i, j) is the value of the j-th eigenfunction at the point x_i. Then we have:

$$\bar{\Phi} = \bar{\mathrm{W}}\hat{\Phi}(n\Lambda)^{-1}. \tag{3.26}$$

Now we are ready to apply the Nyström extension to our clustering problem. Suppose we have a dataset with N examples, where N is very large. We can write the full affinity matrix W for the dataset in the form

$$W = \begin{bmatrix} \mathrm{A} & \mathrm{B} \\ \mathrm{B}^T & \mathrm{C} \end{bmatrix}, \tag{3.27}$$

where A is the affinity matrix for a randomly selected subset of size $n \ll N$ of the entire dataset. By diagonalizing the symmetric matrix A we get

$$\mathrm{A} = \mathrm{U}\Lambda\mathrm{U}^T \tag{3.28}$$

$$\mathrm{AU} = \mathrm{U}\Lambda. \tag{3.29}$$

We can see that (3.29) has the same form as (3.25) with $n = 1$. Thus we can extend the eigenvectors U to the remaining rows of W in the same fashion in which we extended the eigenfunctions to additional points in (3.26). The estimated eigenvectors $\bar{\mathrm{U}}$ of the full affinity matrix W will then be:

$$\hat{\mathrm{U}} = \begin{bmatrix} \mathrm{U} \\ \mathrm{B}^T\mathrm{U}\Lambda^{-1} \end{bmatrix}. \tag{3.30}$$

This is great news: we can estimate the full eigenvectors without computing the very large matrix C in (3.27). However, because in our normalized cuts algorithm we need to scale the rows of the eigenvector matrix by the square roots of the row and column sums, we still need to know the row sums $\boldsymbol{d}$ of W. Fortunately, as was shown in [13], this can be done without computing the full matrix as follows:

$$
\begin{aligned}
\hat{\boldsymbol{d}} &= \hat{\mathrm{W}}\mathbf{1} && (3.31)\\
&= (\hat{\mathrm{U}}\Lambda\hat{\mathrm{U}}^{T})\mathbf{1} && (3.32)\\
&= \begin{bmatrix} \mathrm{U} \\ \mathrm{B}^{T}\mathrm{U}\Lambda^{-1} \end{bmatrix} \Lambda \, [\mathrm{U}^{T} \; \Lambda^{-1}\mathrm{U}^{T}\mathrm{B}]\mathbf{1} && (3.33)\\
&= \begin{bmatrix} \mathrm{U}\Lambda \\ \mathrm{B}^{T}\mathrm{U} \end{bmatrix} [\mathrm{U}^{T} \; \Lambda^{-1}\mathrm{U}^{T}\mathrm{B}]\mathbf{1} && (3.34)\\
&= \begin{bmatrix} \mathrm{A} & \mathrm{B} \\ \mathrm{B}^{T} & \mathrm{B}^{T}\mathrm{A}^{-1}\mathrm{B} \end{bmatrix} \mathbf{1} && (3.35)\\
&= \begin{bmatrix} \boldsymbol{a}_r + \boldsymbol{b}_r \\ \boldsymbol{b}_c + \mathrm{B}^{T}\mathrm{A}^{-1}\boldsymbol{b}_r \end{bmatrix}, && (3.36)
\end{aligned}
$$

where $\boldsymbol{a}_r$ and $\boldsymbol{b}_r$ are the row sums of A and B and $\boldsymbol{b}_c$ are the row sums of B^{T} (i.e. the column sums of B).

3.3.3 Performance

To evaluate the performance of the Nyström method, Fowlkes et al. [13] performed extensive experiments on a synthetically generated dataset with two clusters. They compared the quality of separating the dataset into two clusters (by measuring how far apart the clusters are) when the clustering was performed by solving the dense eigenvector problem versus when the dataset was clustered using the Nyström method for various sizes of the Nyström sample. They found that the quality of the clustering increases monotonically with the size of the Nyström sample. Further, they showed that for a very large real clustering problem (segmentation of an image into groups of pixels), the space spanned by the eigenvectors estimated by the Nyström method was stable for relatively small samples. A similar result was shown by O'Donnell and Westin [29] for a dataset similar to ours—white matter tractography paths in the human brain. In their case, the Nyström method performed well when

used for the separation of a very large set of tractography paths into more than 100 clusters.

3.4 Summary

In this chapter, we gave the background for two groups of unsupervised learning methods: central clustering and spectral grouping. Both types of clustering methods enable us to take a set of examples and group them into a specified number of clusters such that the examples in each cluster are similar to each other and dissimilar from the examples in other clusters. In the following two chapters, we will show how these methods can be applied to sets of observations of moving objects in a scene (Chapter 4) and sequences of such observations (Chapter 5) in order to learn a model that describes the usual pattern of object behavior in the scene.

Chapter 4

Observation-Based Scene Model

In this chapter, we describe a scene model based on individual observations of moving objects. We originally developed this model as a simple but effective way to task a high-resolution pan-tilt-zoom (PTZ) camera in an attentive monitoring system. Figure 4-1 shows a screenshot of a real-time implementation of the system. We use a stationary overview camera with a wide angle of view to collect observations of moving objects in the scene. Every observation is evaluated against a learned scene model, which essentially contains a description of the usual pattern of observations at each scene location. Given the scene model, the PTZ camera is tasked to the most statistically interesting activity, i.e. the observation that is least likely under the model.

We sometimes refer to this approach as a naive scene model because it focuses only on learning the distribution of observation attributes at particular locations in the scene and ignores the sequence in which the observations associated with the motion of an object occured. However, its main advantage is that it is easily learned, can be updated in real time, and is efficient even when many simultaneous activities occur in the scene.

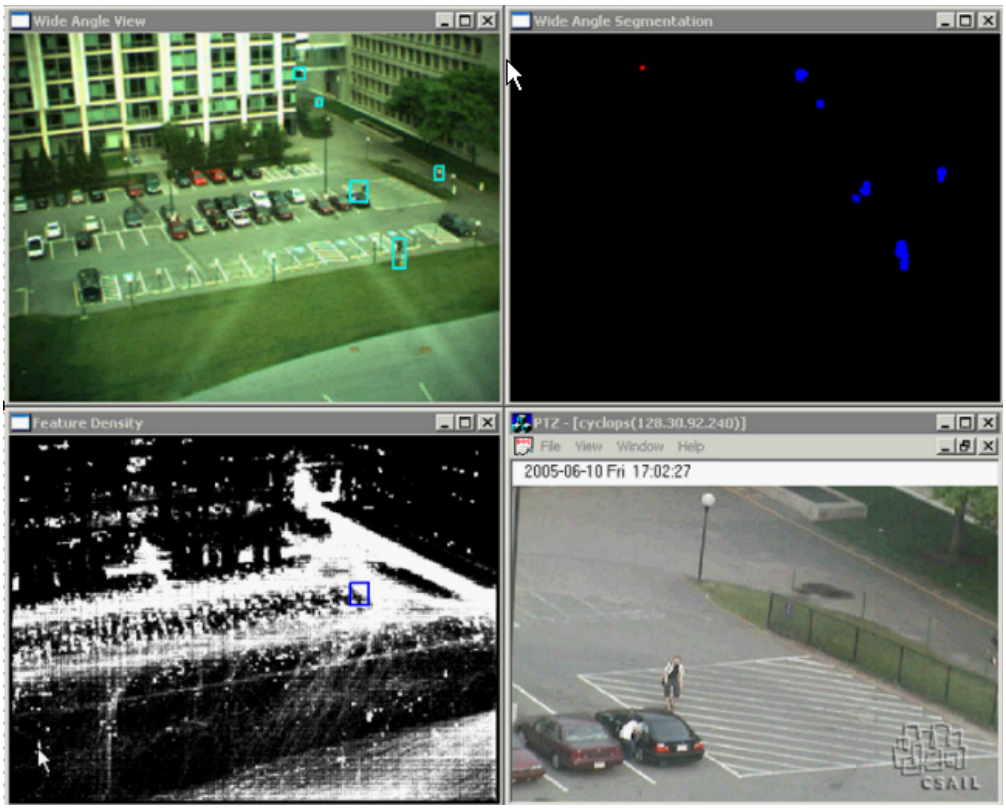


Figure 4-1: Real-time implementation of the observation-based attentive monitoring system. Top left: wide-angle view of the entire scene. Top right: output of the background subtraction algorithm, which serves as the input to the tracking and correspondence algorithm. Bottom left: a visualization of a portion of the observation attribute map. The brighter the pixel, the higher the likelihood of a moving object at that location. The blue rectangle marks the least likely observation currently in the scene. Bottom right: PTZ camera tasked to the marked location.

4.1 Estimating Attribute Maps

Consider a set of observations $O = \{\boldsymbol{o}_i\}$ of moving objects in the scene. Each observation describes the estimated state of an object as it passes through the scene. Various attributes of the state can be estimated, such as position of the centroid, size, velocity, appearance, etc. Let observation $\boldsymbol{o}_i$ be the attribute vector $\boldsymbol{o}_i = [o_i^1..o_i^n]^T$ and let the function $pos(\cdot)$ refer to the position of an observation in image coordinates (we will assume that this is always estimated).

We wish to learn an attribute map $F = \{f_{(x,y)}\}$, in which, at every location (x, y) in image

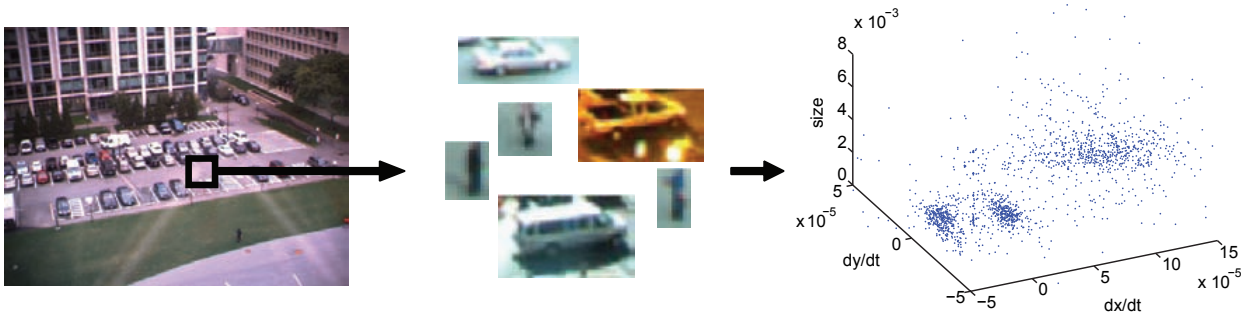


Figure 4-2: Illustration of the attribute map learning framework. Given an image coordinate neighborhood (black rectangle in the left image), we use the history of observations seen in that neighorhood (middle) to learn a mixture of Gaussians in the observation attribute space (right). Here the observation vectors consist of the instantenous velocity ($[\dot{x}, \dot{y}]$) and the size (area of the bounding box). In the distribution on the right, the two tighter clusters correspond to people moving in two major directions and the third cluster corresponds to cars, which are large and only move in one direction through the parking lot.

coordinates, we store the distribution of attributes for all moving objects whose centroids have passed near that location (see Figure 4-2 for an illustration of the process). To do this, we borrow the method used for the dynamic background estimation in [39], where each pixel process in the scene is modeled as a mixture of Gaussians. For simplicity, let us for now assume that the object attributes (the attributes of the tracked blobs) near a particular location in the scene can be modeled as a single multivariate Gaussian. If $\nu(x, y)$ denotes the neighborhood of image location (x, y), then we can model the attributes associated with that neighborhood as a normal distribution with mean $\boldsymbol{\mu}_{(x,y)}$ and covariance matrix $\Sigma_{(x,y)}$ as follows:

$$f_{(x,y)} = \mathcal{N}(\boldsymbol{\mu}_{(x,y)}, \Sigma_{(x,y)}), \tag{4.1}$$

$$\boldsymbol{\mu}_{(x,y)} = mean\{\boldsymbol{o}_i : pos(\boldsymbol{o}_i) \in \nu(x, y)\}, \tag{4.2}$$

$$\Sigma_{(x,y)} = cov\{\boldsymbol{o}_i : pos(\boldsymbol{o}_i) \in \nu(x, y)\}. \tag{4.3}$$

Intuitively, much like modeling pixel processes in background subtraction, we are now

modeling the observation processes in the scene. Thus, we are assuming that observations are independent given the scene model.

4.2 Observation-Based Attention

Given an attribute map $F_{(x,y)}$ as estimated above from the history of observations in a scene, we can now reason about the likelihood of new observations. When a new observation matches the model well, we will consider it typical. If it does not match well, it may warrant further scrutiny as a statistical outlier. Attention at the level of observations thus amounts to thresholding on the likelihood of new observations. We can express the degree to which an observation is anomalous as a function proportional to the negative log likelihood of the observation under the attribute map, as follows:

$$\begin{aligned} anomaly(\boldsymbol{o}) &= -\log p(\boldsymbol{o}|F) && (4.4)\\ &= -\log p(\boldsymbol{o}|f_{pos(\boldsymbol{o})}) && (4.5)\\ &\propto (\boldsymbol{o}-\boldsymbol{\mu}_{pos(\boldsymbol{o})})^T\Sigma^{-1}_{pos(\boldsymbol{o})}(\boldsymbol{o}-\boldsymbol{\mu}_{pos(\boldsymbol{o})}). && (4.6) \end{aligned}$$

There are several possible ways to estimate the attribute map. Given a representative collection of observations, it can be done in batch by simply estimating the parameters of the Gaussian at every image location from the observations that have occured near that location. In our formulation above, we assume that the observations associated with particular image coordinates can be estimated with a single Gaussian; however, the framework can be easily extended to a mixture of Gaussians. In this case, a clustering algorithm such as expectation-maximization or k-means can be used to estimate the parameters.

Alternatively, it is possible to estimate the mixture model parameters incrementally. In the case of Gaussians with diagonal covariance matrices, the mixture model for each attribute included in the attribute map can be estimated using the incremental k-means approximation described in section 2.1.1.

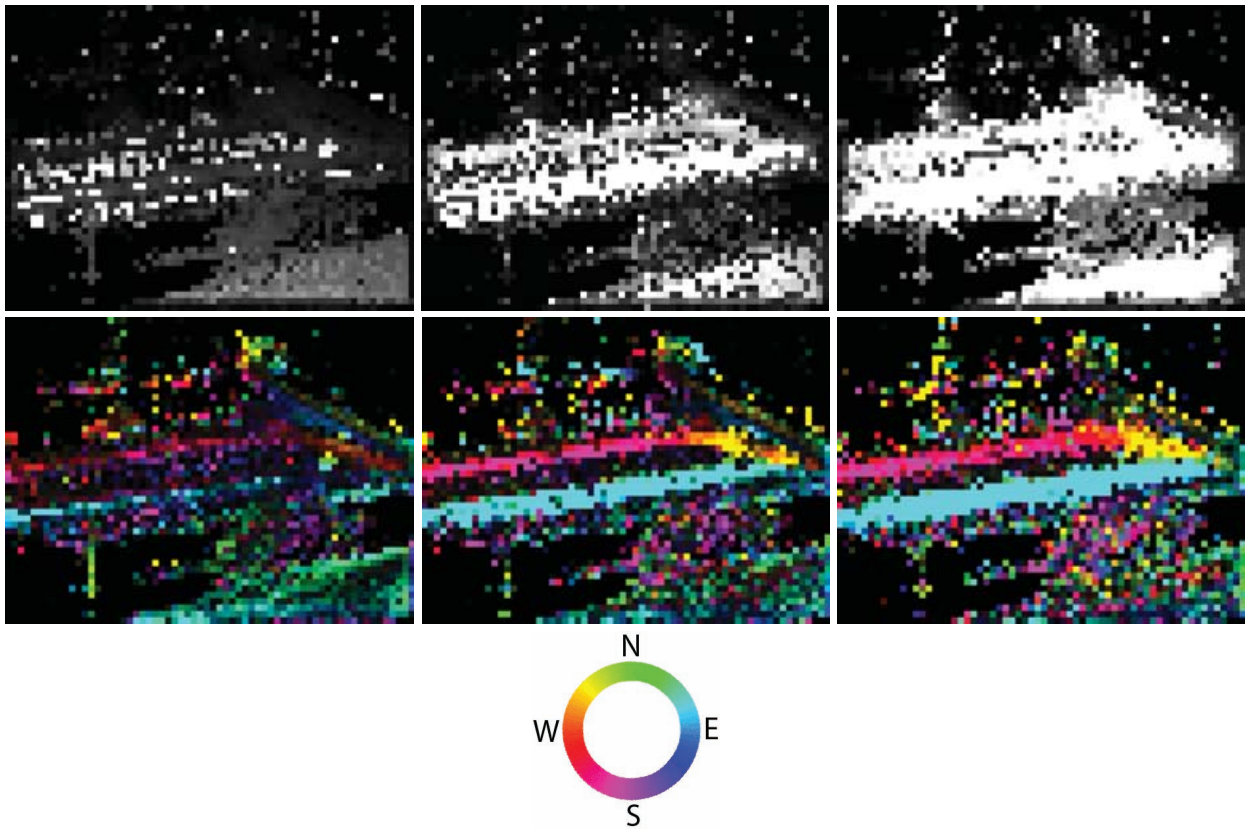


Figure 4-3: Visualization of the first three modes of the attribute maps. Each column corresponds to a mode. In the first column, the brightness in the top image corresponds to the mean object size of the most significant cluster of observations at each location. In the bottom image, the luminance corresponds to the mean speed of motion while the hue shows the mean direction (see the color wheel). The next two columns show the same visualization of the second and third most significant clusters, respectively.

4.3 Results

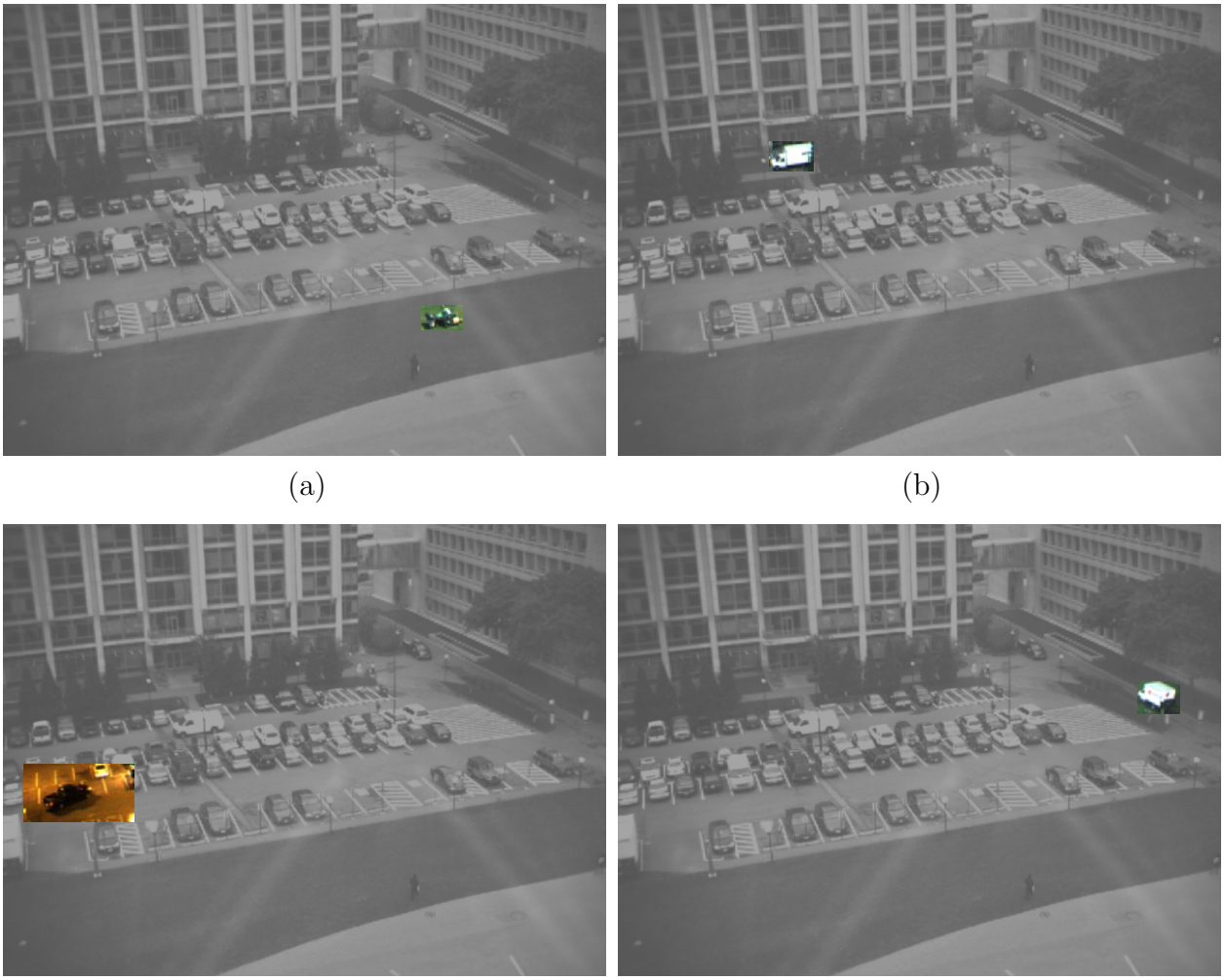
We experimented with a set of observations of moving objects comprising approximately a week of activity in the scene. To estimate the attribute maps, we considered the size (in terms of the area of the bounding box) and velocity of each observation as the attributes of the moving object, i.e. observation $\boldsymbol{o}_i$ was the vector $\boldsymbol{o}_i = [x_i, y_i, \dot{x}_i, \dot{y}_i, s_i]^T$. For each image location along an m x n rectangular grid, we collected all observations whose image coordinates were closest to the grid location. This is akin to binning all of the observations by

their image coordinates into m x n square bins with centers at the grid locations. Given the collection of observations from each bin, we used the k-means algorithm to estimate a mixture of Gaussians over the 3-vectors $[\dot{x}, \dot{y}, s]$. We chose the number of mixture components for each bin to be 4 to represent the cross product of two different object sizes and two different velocities (e.g. cars or groups and pedestrians).

We show a visualization of the resulting observation-based scene model in Figure 4-3. The columns in the figure correspond to the three most significant modes of the mixture distribution (in terms of the number of supporting observations). For each grid location (bin), the top image shows the mean object size, i.e. brighter locations correspond to larger objects. The bottom image shows for each bin the mean velocity shown in HSV color space, where the hue corresponds to the direction of motion (see the color wheel in the figure), the value (brightness) corresponds to the speed and the saturation is constant. For reference, an image of what the scene looks like is shown in Figure 4-1.

Let us examine the three columns of Figure 4-3. The first column shows the most significant mode of the mixture. In most areas of the scene, most of the activity consists of pedestrian traffic. Indeed, we can see that the most significant modes at most locations in the scene have small sizes. The exceptions are the parking spots for cars: most of the activity there consists of cars moving in and out of their spots, i.e. the average object size is large. We can also see that certain areas in the scene have a dominant direction of travel, such as the pedestrian walkways in the upper and lower right of the scene. In contrast, the second most significant modes of the attribute distributions in the parking lot show observations corresponding mostly to cars, which travel at higher speeds and usually move counter-clockwise around the parking lot. The third mode most likely corresponds to groups of people and bicycles in the pedestrian area (larger sizes) and larger cars in the parking lot.

Having built a model that describes the distribution of observations at each location, we can now apply equation (4.6) to detect statistically anomalous observations. Figure 4-4 shows several example detections—observations whose distance from the best matching attribute map centroid was larger than 3 standard deviations. In (a) the lawn mower on



(a) (b)

(c) (d)

Figure 4-4: Examples of detected unusual observations. (a) Lawnmower on the green has both unusual size and direction of motion. (b,d) Trucks in pedestrian zones constitute larger than usual objects. (c) The headlights cause the car to have a very large silhouette.

the grass corresponds to an observation with larger than usual size, traveling in an unusual direction (parallel to the sidewalk). In (b) and (d), the trucks in pedestrian areas are much larger than the usual object. In (c), background subtraction segments out both the car and the light cone emanating from the headlights, resulting in a very large observation.

4.4 Conclusions

An observation-based scene model such as the one we demonstrated in this chapter would be particularly useful in scenes where it is possible to obtain observations of moving objects but difficult to track them as they move through the scene, for instance because the scene is too crowded or there are too many occlusions. In a sense, the model represents a 0th-order description of the motion of moving objects through the scene and is thus a natural extension of the background modeling techniques we described in Section 2.1.1. In adaptive background subtraction, the process that generates particular pixel values at a given image location is estimated from a history of values. Similary, in our observation-based scene model we aim to estimate the process that generates observations of moving objects at a given location. Because we assume these observation processes are independent of each other, the model can be estimated and adapted efficiently in an online fashion using the techniques from adaptive background modeling.

In scenes where it is possible to robustly track moving objects, much additional information can be extracted from the correspondences between observations. For instance, many unusual activities may contain few or no unusual observations. Conversely, an unusual observation is not necessarily indicative of an unusual activity. An example would be a car backing out of a parking lot: even though only a small proportion of observations of cars in the one-way lane of the parking lot have velocities opposite to the normal direction of travel, *sequences* of observations of cars backing out of a parking spot and leaving are quite common.

Nevertheless, many examples of unusual activities do contain unusual observations. Examples include pedestrians in areas where they are not allowed (e.g. the rails in a subway

station) or certain types of vehicles traveling in highway lanes where they are prohibited. The observation-based approach provides a simple and effective way of implementing a basic attention mechanism for an automatic monitoring system in scenes where higher-level descriptions of object motion are not necessary.

Chapter 5

Track-Based Scene Model

In the previous chapter, we described a scene model based on individual moving object observations. In scenes where it is possible to track moving objects, it is desirable to take advantage of the additional information contained in the correspondences and temporal relationships between observations. In this chapter, we develop a scene model and attention framework that is based on moving object tracks. We model how objects move through the scene by clustering a very large set of examples (tracks), and then show how we can classify new tracks using the model as well as determine the degree to which a track is unusual.

5.1 Method Overview

The input to our scene modeling method is a set of object tracks obtained with a background subtraction and tracking algorithm (see chapter 2 for the related background). The output is a scene model that supports classification of new examples and evaluation of the likelihood of examples. The following is an overview of our learning method:

1: Given a set of tracks $T = \{T_1, ..., T_N\}$, we select (uniformly randomly without replacement) a Nyström sample T_s of $n \ll N$ tracks such that $T_s = \{T_{s(1)}, ..., T_{s(n)}\}$.

2: We compute affinities between all pairs of examples (T_i, T_j) such that $T_i \in T_s$ and $T_j \in T$. Distances are computed using a multi-attribute distance measure and converted into

similarities using a multivariate Gaussian kernel with automatically determined weights.

3: We cluster the tracks in T into k clusters using normalized cuts where the spectral embedding of each track is computed from the sample T_s with the Nyström method. We determine the number of clusters k empirically by evaluating the quality of the clustering—we discuss this in more detail in Chapter 6.

4: The scene model is represented as a mixture of Gaussians in the spectral embedding space.

5: New (partial or complete) tracks are projected into the spectral embedding space by comparison to each of the sample tracks in T_s and application of the Nyström extension.

6: The degree to which a track is anomalous is inversely proportional to the log likelihood of the track under the best matching component of the scene model.

In the following sections, we describe each of the above steps in detail, followed by a discussion of results in Chapter 6.

5.2 Track Representation

Throughout this chapter, we assume that we have a system that segments and tracks moving objects in our scene, outputting a set of moving object tracks. Each track is essentially a representation of the object's motion through the scene. Specifically, a track $A = \{\boldsymbol{o}_i^A\}$ is an ordered set of observations describing the estimated state of the object as it moves through the scene (in this case the superscript denotes the track to which the observation belongs and the subscript is the temporal index). The elements of the observation vectors $\boldsymbol{o}_i^A$ describe the object's time-varying attributes, such as size, image coordinates of the centroid, velocity of the centroid, time of day, shape, etc.

5.3 Distance Measure for Tracks

A good distance measure should evaluate two tracks as very similar when the spatial trajectories are collocated in the scene and object attributes such as velocity and size at spatially

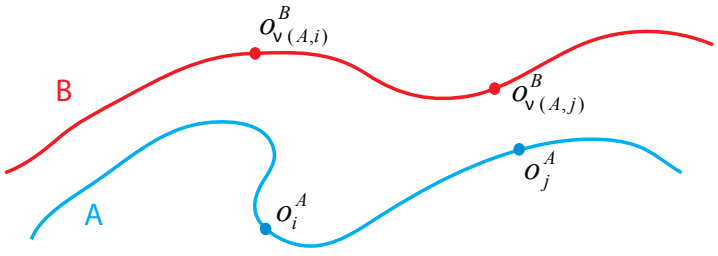


Figure 5-1: Illustration of two track trajectories. Two observations in track A are highlighted along with their spatially corresponding observations in track B.

corresponding points in the tracks are similar. If two objects take the same path but have dramatically different velocities or travel in opposite directions, their tracks should not be similar.

To achieve this, we define the distance between tracks A and B as a vector of attribute-specific distances—one for each object attribute we would like to include in the track comparison:

$$\boldsymbol{D}(A, B) = [d_j(A, B)]. \tag{5.1}$$

To calculate the attribute-specific distance d_j between tracks A and B, we find the average difference in that attribute between observations in track A and their corresponding observations in track B. We use spatial correspondence between observations, i.e. for a given observation in A, the corresponding observation in B is the closest one in image coordinates. The intuition behind this choice is that if two paths are similar, spatially proximal observations along the two paths will have similar attributes. Note, however, that through our choice of attributes we can still use temporal information in the tracks. For instance, the track of a car that drives through the parking lot without stopping will be different from the track of a car that stops or slows down before continuing even if the trajectories are identical because the velocities of spatially corresponding observations will differ.

Let $\text{pos}(\boldsymbol{o}^A_i)$ be the image position of the observation $\boldsymbol{o}^A_i$ in track A, and $\boldsymbol{o}^B_{\nu(A,i)}$ be the observation in track B whose position is closest to $\text{pos}(\boldsymbol{o}^A_i)$, as illustrated in Figure 5-1:

$$\boldsymbol{o}^B_{\nu(A,i)} = \underset{\boldsymbol{o}^B_j}{\operatorname{argmin}} \|\operatorname{pos}(\boldsymbol{o}^A_i) - \operatorname{pos}(\boldsymbol{o}^B_j)\|. \tag{5.2}$$

The directed attribute-specific distance between tracks for the j-th attribute is:

$$d_j(A \to B) = \frac{1}{|A|} \sum_i d(\boldsymbol{o}^A_i(j), \boldsymbol{o}^B_{\nu(A,i)}(j)), \tag{5.3}$$

and the undirected (symmetric) distance is:

$$d_j(A, B) = \min(d_j(A \to B), d_j(B \to A)). \tag{5.4}$$

The distance in (5.3) is very similar to the Hausdorff distance, except that instead of finding the largest distance between pairs of closest observations, we take the average over the distances between closest observations along the entire track. Using the average rather than the maximum as the statistic makes the distance measure much more robust to tracking errors. We use the minimum to symmetrize the distance, as in [47], because we do not assume that tracks have equal numbers of observations or that they are complete. When a full track is compared to a partial track along the same path, as long as the corresponding observations have similar attributes, we want the partial track to be clustered together with the full track. Thus we use the smaller of the two directed distances.

The function $d(\boldsymbol{o}_1(j), \boldsymbol{o}_2(j))$ in equation (5.3) is simply the scalar difference in the j-th attribute between the two observations. For instance, if the attribute is size, it is the difference between the sizes of the objects. All distances are squared when we convert the distance between tracks to a similarity, so the sign does not matter.

Note that the distance measure described in this section is not the only possible choice—other distance measures between sequences of moving observations could be used; several are discussed in Section 2.4.1. We chose the modified Hausdorff vector distance described above because it has the following important characteristics: (1) it is scalable with respect to the moving object attributes used for the comparison and (2) it does not require manual setting of any parameters (unlike some other distance measures such as the least common

subsequence or the Euclidean distance in the space of hidden Markov models—see Section 2.4.1). Additionally, the modified Hausdorff distance has been shown to perform well in similar clustering problems [47, 29].

In our experiments (described in Chapter 6), we use the size of the silhouette bounding box, the speed, direction of motion, position in the image and time of day as the object attributes. Other attributes such as object shape or appearance can also be incorporated.

5.4 Converting Distances to Similarities

Our spectral clustering algorithm requires an affinity matrix, so we must convert distances between examples to similarities. We convert the distances $\boldsymbol{D}$ from (5.1) to similarities with a multivariate Gaussian kernel as follows:

$$S(A, B) = \exp(-\boldsymbol{D}^T(A, B)\Sigma^{-1}\boldsymbol{D}(A, B)). \tag{5.5}$$

Similar Gaussian kernel methods (but for 1-dimensional distances) have been used in many other clustering approaches [37, 28, 30, 48, 29, 47]. If we consider the covariance matrix Σ to be diagonal, i.e., $\Sigma = \text{diag}(\sigma_j^2)$, substituting (5.1) into the equation above yields:

$$S(A, B) = \exp\left(-\sum_j \frac{d_j^2}{\sigma_j^2}\right) \tag{5.6}$$

$$= \prod_j \exp\left(-\frac{d_j^2}{\sigma_j^2}\right), \tag{5.7}$$

where d_j is the attribute specific distance for the j-th attribute. Thus for a diagonal kernel, the resulting similarity is a product of the similarities for each attribute with the variances σ_j^2 serving essentially as the weighting factors. This is intuitively satisfying—we want two tracks to be similar if and only if they are similar for all of the attributes. For this reason

and for simplicity (the diagonal matrix has fewer parameters), in our experiments we use a diagonal kernel. However, the methods discussed in this section would also apply to a kernel with a full covariance matrix, which may be useful in scenes where different object attributes are strongly correlated.

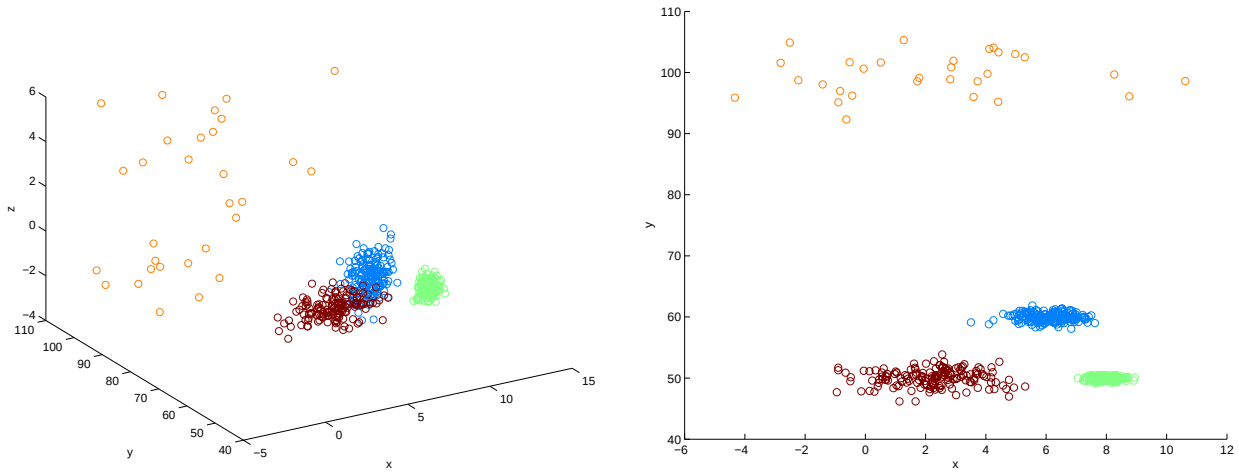
There has been some discussion as to how to best set the parameters (variances) in the kernel Σ. Some approaches suggest running the clustering several times with different parameters in order to determine the best values (e.g. [28]), which dramatically increases the computational complexity. Others have shown that setting the parameters separately for each example based on the local structure of the similarity space leads to better clustering results [12, 55]. In practice, the parameters are usually tuned by hand—an exercise in patience that must be repeated for each new dataset.

We suggest an automatic method to tune the parameters in Σ similar to the one proposed by Zelnik-Manor and Perona [55]. They convert the 1D distance $d(x, y)$ between examples x and y into a similarity by setting separate variances σ_x, σ_y for each example and then setting the similarity as follows

$$S(x, y) = \exp\left(-\frac{d^2(x, y)}{\sigma_x \sigma_y}\right). \tag{5.8}$$

The local parameter σ_x is set to the distance between x and its K-th nearest neighbor, where K depends on the dataset. Similarly, σ_y is set to the distance between y and its K-th nearest neighbor. The intuition behind this method is that σ_x bears a relationship to the local variance of the data as "seen" from point x, whereas σ_y is the local variance as "seen" from point y. Thus to compare the two points, a Gaussian kernel with variance equal to the product of the two local variances is used. The same method can be applied to multidimensional distances by calculating the similarities along each dimension and taking their product to obtain the final similarity, as in equation (5.7).

Our method does not involve any additional parameters and is based on the definition of variance as a mean squared difference. The intuition, however, remains the same: the variance of the Gaussian kernel is set for each point based on the variance of the data as "seen" from that point. We set $\Sigma_{A,B}$ for each pair of tracks separately by setting the diagonal



(a) (b)

Figure 5-2: Synthetic dataset generated from a 3D Gaussian mixture with 4 components, shown from two different angles. Note that the four mixture components have dramatically different priors and variances.

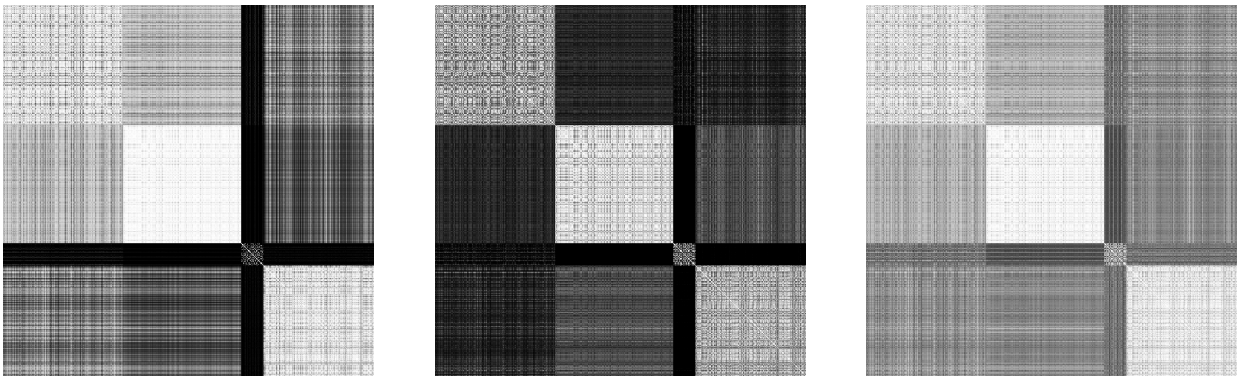
elements to be the average squared distance from A or B to the rest of the tracks in the Nyström sample T_s, whichever is larger:

$$\Sigma_{A,B} = \text{diag}(\sigma_j^2(A,B)), \tag{5.9}$$

$$\sigma_j^2(A,B) = \max(\frac{1}{n}\sum_k d_j^2(A,T_{s(k)}), \frac{1}{n}\sum_k d_j^2(B,T_{s(k)})). \tag{5.10}$$

As a reminder, the subscript j above refers to the attribute, e.g., size or velocity, of observations in the track. We use the larger of the variances for each pair of points to preserve the similarity information even between points further away from each other. In our experience, this leads to more stable performance of the Nyström method.

Consider the synthetic dataset in Figures 5-2(a-b). The 500 points in the dataset have a Gaussian mixture distribution with four components, each with a different mean, covariance matrix and prior. Figure 5-3(a-c) shows the affinity matrices for this data calculated using three different methods. In (a), the same diagonal covariance matrix was used for each example. In (b) the local variance method from equation (5.8) was used and (c) shows the

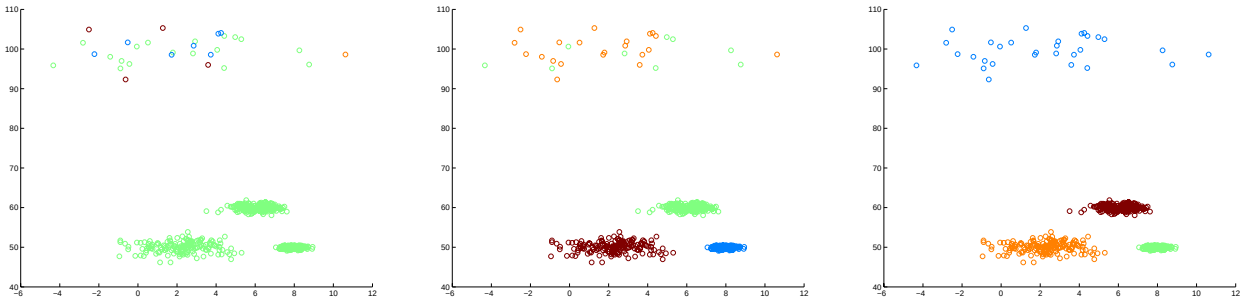


(a) (b) (c)

Figure 5-3: Affinity matrices for the synthetic dataset in Figure 5-2 calculated by (a) using a global Gaussian kernel, (b) estimating local kernel parameters using the method from [55] (equation (5.8)) and (3) using our parameter-free local method. The rows and columns in the affinity matrices are ordered by the ground truth. Higher grayscale values indicate larger pairwise similarities.

result of our parameter-free local variance method. In (a) and (b) we chose the parameter values that most amplify the block-diagonal structure of the matrix. The global method leads to a poor affinity matrix, in which one of the clusters has very small within-cluster affinities. Our method gives a similar (but somewhat flatter) affinity matrix to the method in (b) but without the need to set a parameter. Additionally, in our experiments with the Nyström method, we have observed that if the sample affinity matrix contains rows with very small row sums, the estimation of the row sums of the full affinity matrix leads to numerical instabilities, sometimes resulting in negative row sum estimates. Our method of converting distances to similarities tends to lead to "smoother" affinity matrices and thus seems better suited for the Nyström extension.

In Figure 5-4 we show sample clustering results for the synthetic dataset using the three affinity matrices from Figure 5-3. We used the normalized cuts algorithm from the previous chapter (Algorithm 3.3.1) and we used the leading 4 eigenvectors for the spectral embedding. The k-means stage of the algorithm was run 10 times and the best run (in terms of mean distance to nearest centroid) was selected as the final clustering. The global variance setting does not lead to a good solution. To see why, let us again examine the data set shown in



(a) (b) (c)

Figure 5-4: The synthetic dataset clustered using the normalized cuts algorithm with the three affinity matrices from Figure 5-3: (a) global variances, (b) local variances using (5.8) and (3) local variances using our method. Colors correspond to the cluster assignment (and may be permuted). The correct clustering is shown in Figure 5-2(b).

Figure 5-2. In order for the points in the most spread-out cluster to appear similar to one another, the variance must be large. Unfortunately, the three tighter clusters then appear very similar to each other. Conversely, for a variance setting that separates the three tight clusters from each other, the points in the fourth cluster appear very dissimilar from each other. Thus, no single variance setting is optimal for such a dataset. Setting the variances locally for each data point enables us to capture the local similarity relationships. It is clear from these observations that the Zelnik-Manor and Perona method as well as our local variance method both result in a better clustering.

5.5 Model Learning

We now have all the tools to learn a track-based scene model by clustering moving object tracks. We perform the spectral clustering algorithm from section 3.3 using the Nyström method to estimate the eigenvectors of the affinity matrix from the sample of tracks T_s. We summarize all of the steps in Algorithm 5.1.

The algorithm essentially projects each track into the spectral embedding space, which itself characterizes the principal components of the similarity space, as we discused in section 3.3. Once mapped into the spectral embedding space, the tracks are then clustered using a

Algorithm 5.1 Learning a track-based scene model

Input: Set of tracks T; number of clusters k; dimension of embedding space $N_E \ll N$; size of Nyström sample n.

Output: Gaussian mixture model M, $M_i = (\boldsymbol{\mu}_i, \Sigma_i, \pi_i)$.

1: From T, select the Nyström sample of tracks T_s.

2: Using the distance measure from section 5.3, calculate the pairwise distances $\boldsymbol{D}(A, B)$ for all pairs of tracks (A, B) s.t. $A \in T_s, B \in T$.

3: Convert the distance vectors $\boldsymbol{D}$ to affinities using equation (5.10). This yields the matrices A and B in the full affinity matrix $W = \begin{bmatrix} \mathrm{A} & \mathrm{B} \\ \mathrm{B}^T & \mathrm{C} \end{bmatrix}$.

4: Calculate the row sums $\boldsymbol{a}_r = \mathrm{A}\mathbf{1}, \boldsymbol{b}_r = \mathrm{B}\mathbf{1}$ and the column sums $\boldsymbol{b}_c = \mathrm{B}^T\mathbf{1}$.

5: Estimate the row sums $\hat{\boldsymbol{d}}$ of W as: $\hat{\boldsymbol{d}} = \begin{bmatrix} \boldsymbol{a}_r + \boldsymbol{b}_r \\ \boldsymbol{b}_c + \mathrm{B}^T\mathrm{A}^{-1}\boldsymbol{b}_r \end{bmatrix}$. Note that W is symmetric, i.e. $\hat{\boldsymbol{d}}$ are also the column sum estimates of W.

6: Scale A: $\mathrm{A}' = (a'_{ij})$ where $a'_{ij} = a_{ij}/\sqrt{\hat{d}_i\hat{d}_j}$.

7: Scale B: $\mathrm{B}' = (b'_{ij})$ where $b'_{ij} = b_{ij}/\sqrt{\hat{d}_i\hat{d}_j}$.

8: Diagonalize A′: $\mathrm{A}' = \mathrm{U}\Lambda\mathrm{U}^T$.

9: Apply the Nyström extension: $\hat{\mathrm{U}} = \begin{bmatrix} \mathrm{U} \\ \mathrm{B}^T\mathrm{U}\Lambda^{-1} \end{bmatrix}$.

10: Scale U: $\mathrm{U}' = (u'_{ij})$ where $u'_{ij} = u_{ij}/\sqrt{\hat{d}_i}$.

11: $\mathrm{U}_E \leftarrow$ columns 2 through $N_E + 1$ of U′.

12: The embedding vectors for the data are $X \leftarrow \{\boldsymbol{x}_1, ..., \boldsymbol{x}_{N_E}\}$ where $\boldsymbol{x}_i$ is the i-th row of U_E.

13: Cluster X using k-means or EM to obtain clusters $C_i, i \in \{1, ..., k\}$.

$$\boldsymbol{\mu}_i = \frac{1}{|C_i|} \sum_{\boldsymbol{x}_j \in C_i} \boldsymbol{x}_j \tag{5.11}$$

$$\Sigma_i = \frac{1}{|C_i| - 1} \sum_{\boldsymbol{x}_j \in C_i} (\boldsymbol{x}_j - \boldsymbol{\mu}_i)(\boldsymbol{x}_j - \boldsymbol{\mu}_i)^T \tag{5.12}$$

$$\pi_i = \frac{|C_i|}{\sum_{j=1}^{k} |C_j|}. \tag{5.13}$$

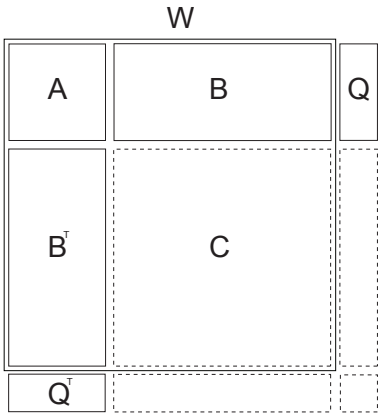


Figure 5-5: Illustration of the relationship between the affinity matrices used for learning (W) and for classification (Q). The matrix Q corresponds to the similarities between new examples and the Nyström sample. A is the affinity matrix of the Nyström sample, while B are the pairwise affinities between the rest of the learning set and the Nyström sample. W is the full affinity matrix of the learning set. The matrix segments drawn in dashed lines are not computed.

central clustering method such as k-means or EM. Thus the logical next step is to represent the resulting clusters of tracks as Gaussians in the embedding space. The inspiration for this representation came from [29], where a similar framework was used to build an anatomical atlas of brain white matter tractography.

5.6 Classification

Having defined a mixture model over our set of examples in the spectral embedding space, we can now classify new examples by finding the best matching component of the mixture model. We use the method from [29] to project new examples into the spectral embedding space. Let $T_n = \{T_{n(1)}, ..., T_{n(p)}\}$ be a new set of examples. We first calculate the n x p matrix Q $= (q_{ij})$ where $q_{ij} = s(T_{s(i)}, T_{n(j)})$ is the similarity between the track $T_{s(i)}$ from the Nyström sample and $T_{n(j)}$ from the new set. Figure 5-5 shows how this new affinity matrix Q is related to the affinity matrix W of the data used for learning the model. We now need

to scale each entry in the matrix Q as we did with the affinity matrices of the learning set. We want to perform the scaling in such a way that if a new example is identical to one that occured in the learning set, its embedding will be the same. To do this, we will assume, as in [29] that because a large number of examples were used for learning the model, adding the additional columns Q to the affinity matrix W will not significantly increase the row sums of W. Thus, we will use the original row sums $\hat{\boldsymbol{d}}$ for the scaling. That leaves us with calculating the column sums for the scaling of Q. We use the same method as we did in the learning (see step 5 of Algorithm 5.1) to extend the column sums of Q and obtain the estimated full column sums $\hat{\boldsymbol{r}}$ as follows:

$$\hat{\boldsymbol{r}} = \mathrm{Q}^T\mathbf{1} + \mathrm{Q}^T\mathrm{A}^{-1}\boldsymbol{b}_r. \tag{5.14}$$

Now we can scale each element in Q by the square root of the product of the estimated row and column sums. The scaled version of Q is:

$$\mathrm{Q}' = (q'_{ij}), \text{ where } q'_{ij} = q_{ij}/\sqrt{\hat{d}_i\hat{r}_j}. \tag{5.15}$$

We can now again apply the Nyström method to extend the embedding space eigenvectors onto these new points as follows:

$$\bar{\mathrm{U}} = \mathrm{Q}'^T\mathrm{U}\Lambda^{-1}. \tag{5.16}$$

The remaining final step is to now scale the rows of the embedding matrix $\bar{\mathrm{U}}' = (\bar{u}'_{ij})$:

$$\bar{u}'_{ij} = \bar{u}_{ij}/\sqrt{\hat{d}_i}. \tag{5.17}$$

The i-th row of the scaled embedding matrix $\bar{\mathrm{U}}'$ is thus the embedding vector for the i-th example from the new set T_n. Given an embedding vector $\boldsymbol{e}$, we can now find its class label

$c(\boldsymbol{e})$ by finding the best matching component of our model:

$$\begin{aligned} c(\boldsymbol{e}) &= \underset{i}{\operatorname{argmax}} \Pr(C_i|\boldsymbol{e}) && (5.18) \\ &= \underset{i}{\operatorname{argmax}} \frac{p(\boldsymbol{e}|C_i)\Pr(C_i)}{\sum_j p(\boldsymbol{e}|C_j)\Pr(C_j)}, && (5.19) \end{aligned}$$

where $p(\boldsymbol{e}|C_i)$ is the likelihood of the embedding vector $\boldsymbol{e}$ under the mixture model and $\Pr(C_i) = \pi_i$ is the mixture prior. Ignoring constant terms and taking the log of the likelihoods yields the following quadratic classifier:

$$c(\boldsymbol{e}) = \underset{i}{\operatorname{argmax}} \left(-\frac{1}{2}\log|\Sigma_i| - \frac{1}{2}(\boldsymbol{e}-\boldsymbol{\mu}_i)^T\Sigma^{-1}(\boldsymbol{e}-\boldsymbol{\mu}_i) + \log\pi_i \right). \quad (5.20)$$

5.7 Anomaly Detection

Given our probabilistic scene model framework, determining the degree to which a new example is anomalous is very similar to the classification process. The better an example matches the model, the less statistically unusual it is. Thus given a track T_i with spectral embedding $\boldsymbol{e}_i$, its *anomaly* can be quantified as:

$$anomaly(T_i) = -\log \sum_i p(\boldsymbol{e}|C_i)\pi_i. \quad (5.21)$$

For simplification, we can make the assumption that $p(\boldsymbol{e}|C_i) = 0$ whenever $c(\boldsymbol{e}) \neq i$. Then the anomaly becomes:

$$\begin{aligned} anomaly(T_i) &= -\log\left(p(\boldsymbol{e}|C_{c(\boldsymbol{e})})\pi_{c(\boldsymbol{e})}\right) && (5.22) \\ &= \frac{1}{2}\log|\Sigma_{c(\boldsymbol{e})}| + \frac{1}{2}(\boldsymbol{e}-\boldsymbol{\mu}_{c(\boldsymbol{e})})^T\Sigma^{-1}_{c(\boldsymbol{e})}(\boldsymbol{e}-\boldsymbol{\mu}_{c(\boldsymbol{e})}) - \log\pi_{c(\boldsymbol{e})}. && (5.23) \end{aligned}$$

In the discussion of results in the following chapter, we will refer to the above quantity as the statistical anomaly score for a given track.

5.8 Adaptive Scene Modeling

Thus far we have described our track-based scene modeling algorithm as a batch process: given a large set of examples, we learn a model of the moving object activity in the scene. An implicit assumption is that the distribution of tracks is stationary: given enough data, once the model is learned, it can be used indefinitely. However, in certain scenes, this may not be a good long-term assumption. For instance, a construction project might alter the usual paths of vehicles and pedestrians in a scene, causing many new detections to be unlikely under the learned model. The simplest way to approach this problem is to re-learn the scene model whenever the long-term rate of anomalous activity rises above a certain predefined threshold.

However, it may also be possible to adapt the scene model online using newly acquired evidence. To do so, we could take advantage of the ability to project new examples into the same spectral embedding space in which the model was learned. As a reminder, the scene model is a mixture of Gaussians, in which each component (cluster), specified by its mean and covariance matrix in the embedding space, corresponds to a pattern of activity in the scene. Given a new example, we could update the parameters of the best-matching Gaussian or, if there is no good match (e.g. if the Mahalanobis distance of the example from the nearest centroid is above some threshold), a new cluster could be added to the model. The update equations would be similar to those used in adaptive background subtraction in Section 2.1.1.

The adaptive method does not require repeating the expensive clustering step, but it comes with a caveat: it assumes that the embedding space estimated from the original Nyström sample remains to be applicable to the current activity in the scene. Developing methods to determine whether the spectral embedding space remains to be a good representation of the similarity space of a new set of examples is an intriguing future direction of research.

5.9 Surprise Detection

In addition to classification and anomaly detection, we can also take advantage of the fact that our track distance measure allows us to classify partial tracks. Let T_i^t be a track in which we have observations from time 0 (the beginning of the track) up to time t and let $\boldsymbol{e}_i^t$ be the corresponding spectral embedding. We can now maintain a belief distribution $\Pr(C_i|\boldsymbol{e}_i^t)$ that describes how well the track up to time t matches each component of our mixture model and we can use the notion of computable surprise [19, 18] to detect surprising moments in the development of a track.

Computable surprise, as introduced by Itti and Baldi [19], is an information-theoretic way to determine how "surprising" new data is in terms of how much it changes the observer's beliefs. Suppose we have some set of models $\mathcal{M}$ and we receive some new data D. We can measure the impact of the data, i.e. the degree to which D is surprising to the observer, by measuring how much it changes our belief from the priors $\Pr(M)$ where $M \in \mathcal{M}$ to the posteriors $\Pr(M|D)$. The change, i.e., the surprise $S(D)$ associated with the new data, can be measured by the Kullback-Leibler divergence (relative entropy) as follows:

$$S(D) = \int_{\mathcal{M}} \Pr(M) \log \frac{\Pr(M)}{\Pr(M|D)} dM. \tag{5.24}$$

Relating this to the problem of detecting a surprising moment in a developing track, suppose that we have a current belief distribution over our set of clusters C_j and we obtain some new observations of our tracked object. We will consider these new observations surprising if our belief over the track, i.e., the distribution over the mixture components in our model, significantly changes as a result of obtaining the new observations. The surprise associated with the track T_i at time $t+1$ given the track at time t then becomes:

$$S(T_i^{t+1}|T_i^t) = \sum_j \Pr(C_j|T_i^t) \log \frac{\Pr(C_j|T_i^t)}{\Pr(C_j|T_i^{t+1})}. \tag{5.25}$$

Intuitively, the above quantity will be large whenever a new set of observations for a developing track causes a large change in the posterior distribution over the mixture compo-

nents of our model. If we evaluate the surprise at regular time intervals as the track develops, a large surprise value will indicate that the latest set of observations contained a surprising moment in the track. We show some experimental results in Chapter 6.

5.10 Summary

We have shown how spectral clustering along with the Nyström method can be applied to tracking data in order to build a track-based scene model. In the model, we represent groups of tracks (activities) as Gaussians in the spectral embedding space. We can then evaluate new examples in terms of their likelihood under the mixture model, which results in a statistical anomaly score for each example. In the context of an attentive monitoring system, the anomaly score can be used to determine where to direct the attention of a high-resolution sensor in order to collect more detailed information about unusual activities. Additionally, the model allows us to reason about surprising moments along developing tracks. Having thus established the framework for a track-based statistical attention model, we now turn to evaluating the performance of such a model on real data.

Chapter 6

Results and Evaluation

Using the methods from the previous chapters, we are finally ready to test the performance of our scene-model learning framework on real data collected from a busy urban outdoor scene. In this chapter, we show the results of applying our clustering algorithm to a week of tracking data, mining the data for unusual events and comparing our anomaly detection framework with human perception of what is unusual by obtaining human judgments on a set of examples from our dataset.

6.1 Data Collection

To evaluate our algorithms, we collected a week of tracking data from a busy campus parking lot scene. The data was collected using a stationary camera placed in a window on the 8th floor of an office building adjacent to the parking lot. Images captured by the camera at 15 frames per second, 24 hours per day, were processed by our background segmentation and tracking systems, which output a collection of moving object tracks. Because the goal was to mine the data for unusual activity, we limited the pre-processing to the following:

1. We used simple heuristics to exclude tracks that were too short (i.e. less than 5 seconds) or most likely due to noise in the scene. Because the goal was to detect unusual activity, this filtering step was extremely conservative (for details, see Appendix A).

After filtering, we were left with a set of nearly 40,000 tracks.

2. As tracking was performed at 15 frames per second and tracks may be several minutes long, many tracks had more than 1,000 observations. To reduce the computational complexity, we pruned the tracks to contain observations at least 500ms apart. The pruning was done by simply omitting observations that occurred less than 500ms after the last included observation, where the first included observation was the first observation of the track.

6.2 Clustering

We clustered the set of roughly 40,000 tracks using the approximate spectral clustering algorithm described in Chapter 5. We randomly selected 3,000 tracks to serve as the Nyström sample (the sample size was limited by the largest eigenvalue problem that fit into memory in our MATLAB implementation). Pairwise distances between tracks in the sample were computed using the similarity measure from section 5.4.

We performed two sets of experiments. In the first, we compared tracks in terms of the following attributes: position ($[x, y]$ in image coordinates), size of the object (the area of the bounding box)[1], the instantaneous speed ($[\dot{x}, \dot{y}]$), and the direction of motion represented as an angle between 0 and 2π. In the second experiment, we added time of day as an additional attribute.

In addition to the Nyström sample size, our clustering algorithm requires choosing two additional parameters: number of clusters k and dimensionality of the embedding space N_E. In the absence of ground truth, we ran the clustering algorithm multiple times for various values of these parameters. In order for such a parameter search to be effective, some measure of "goodness" of the clustering is needed. Though no measure is definitive, intuitively a good clustering is one in which the clusters are tight and well separated. The standard way to

[1] In a far-field setting, where the sizes of the observed objects are much smaller than the distance to the camera, the bounding box area is a good representation of the object size, as most objects appear merely as rigid blobs. In a near-field setting, where the articulation and detailed shape of objects can be observed, more descriptive attributes such as the shape or area of the silhouette might be useful.

evaluate the tightness of clusters is to measure the mean squared distance to cluster centroid for all examples (essentially this is the mean squared error). Tighter clusters will have a smaller mean squared distance to the centroid (note that this quantity will also decrease with the number of clusters). At the same time, clusters should also be well separated, i.e. the mean squared distance between means of neighboring clusters should be large.

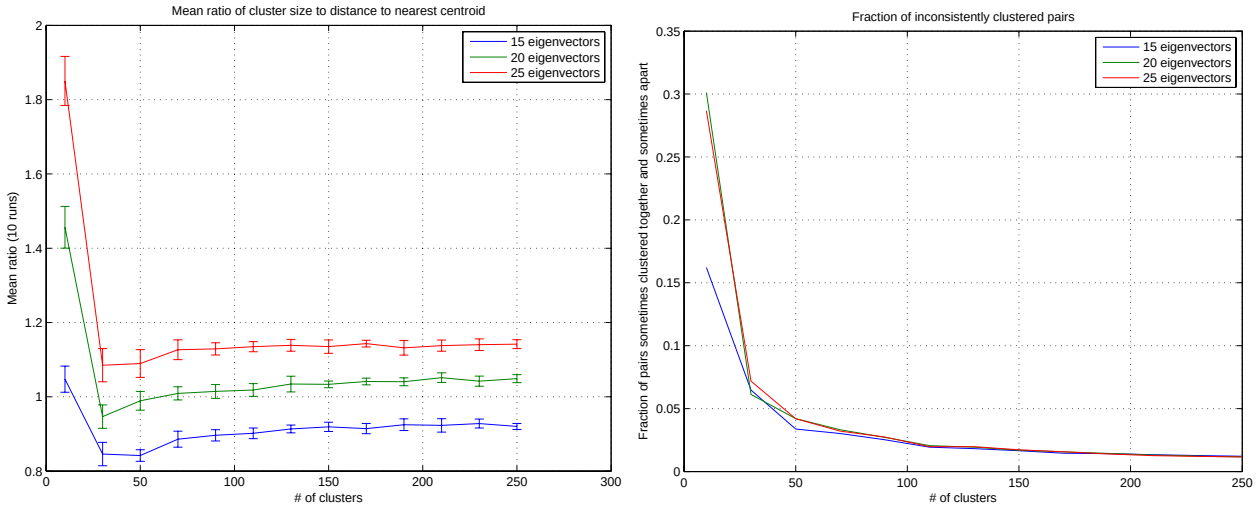
Thus for each cluster we evaluate the ratio of a cluster's size (as measured by the mean squared distance to the centroid) to the squared distance between the cluster's centroid and the centroid of the closest cluster. Taking the mean of this ratio over all clusters for a clustering C leads to the cluster quality measure $Q(C)$:

$$Q(C) = \frac{1}{k} \sum_{j=1}^{k} \frac{\frac{1}{|C_j|} \sum_{\boldsymbol{e} \in C_j} (\boldsymbol{e} - \boldsymbol{\mu}_j)^2}{\min_i (\boldsymbol{\mu}_j - \boldsymbol{\mu}_i)^2}. \tag{6.1}$$

As the number of clusters increases, this ratio should eventually stabilize when the intrinsic clusters in the data begin to be subdivided.

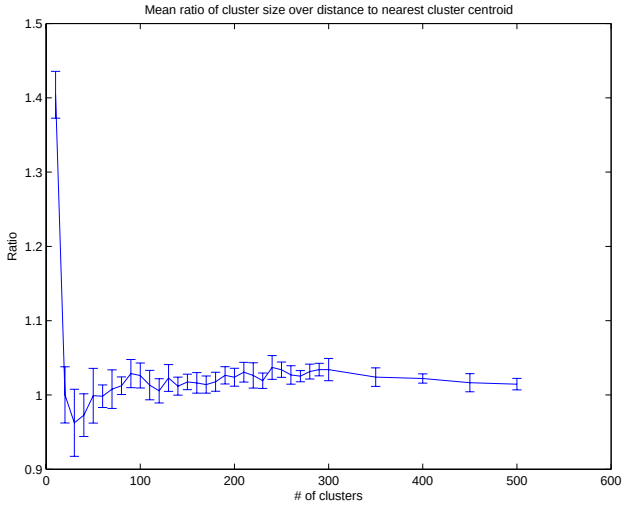
Another way of measuring the quality of clustering was suggested by O'Donnell and Westin [29]. They suggest that an appropriate number of clusters is one for which the fraction of inconsistently clustered example pairs over several runs of the algorithm is small. Pairs of examples that are sometimes clustered together and sometimes apart over different runs of the algorithm are considered inconsistent.

We performed an experiment in which we clustered the dataset using various numbers of clusters k for three different dimensionalities of the embedding space. For each pair of values k and N_E, we ran the k-means stage of the spectral clustering algorithm 10 times. In Figure 6-1(a) we plot the relationship between the clustering criterion Q defined above and the number of clusters. Figure 6-1(b) shows for each k the estimated fraction of example pairs that were clustered together in some runs of the algorithm and apart in others. Because of the size of our dataset, rather than considering all pairs of examples, we calculate this measure for all pairs of examples from a randomly selected subset of the data. Figure 6-1(c)-(d) shows both experiments for embedding dimensionality $N_E = 20$ with the number of clusters sampled at finer intervals. The results suggest that the dimensionality of the embedding space does not

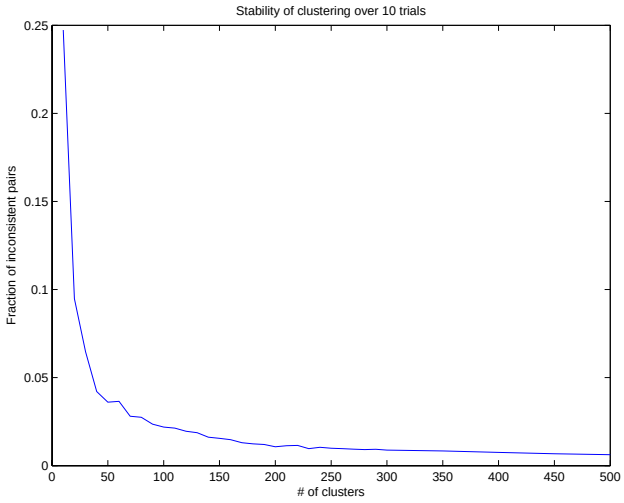


(a)

(b)



(c)



(d)

Figure 6-1: Different methods of evaluating the clustering quality as a function of the number of clusters. In (a) we show the mean ratio of cluster size to distance to the nearest cluster (see equation 6.1 in text). In (b) we show the fraction of inconsistently clustered pairs of examples. (c,d) show the the experiments from (a,b) for 20 eigenvectors with the number of clusters sampled at finer intervals. For each number of clusters, we ran the k-means stage of the spectral clustering algorithm 10 times.

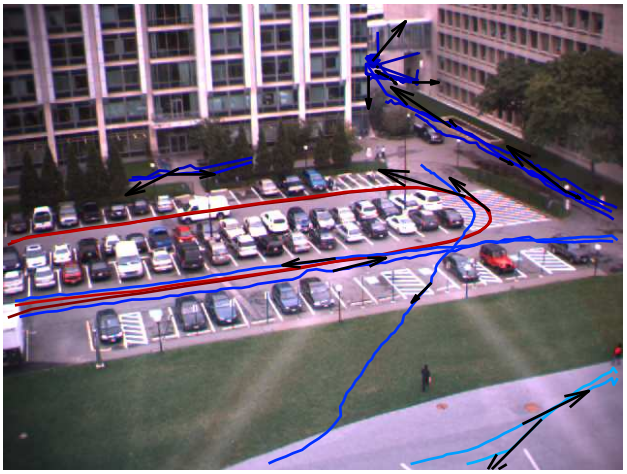
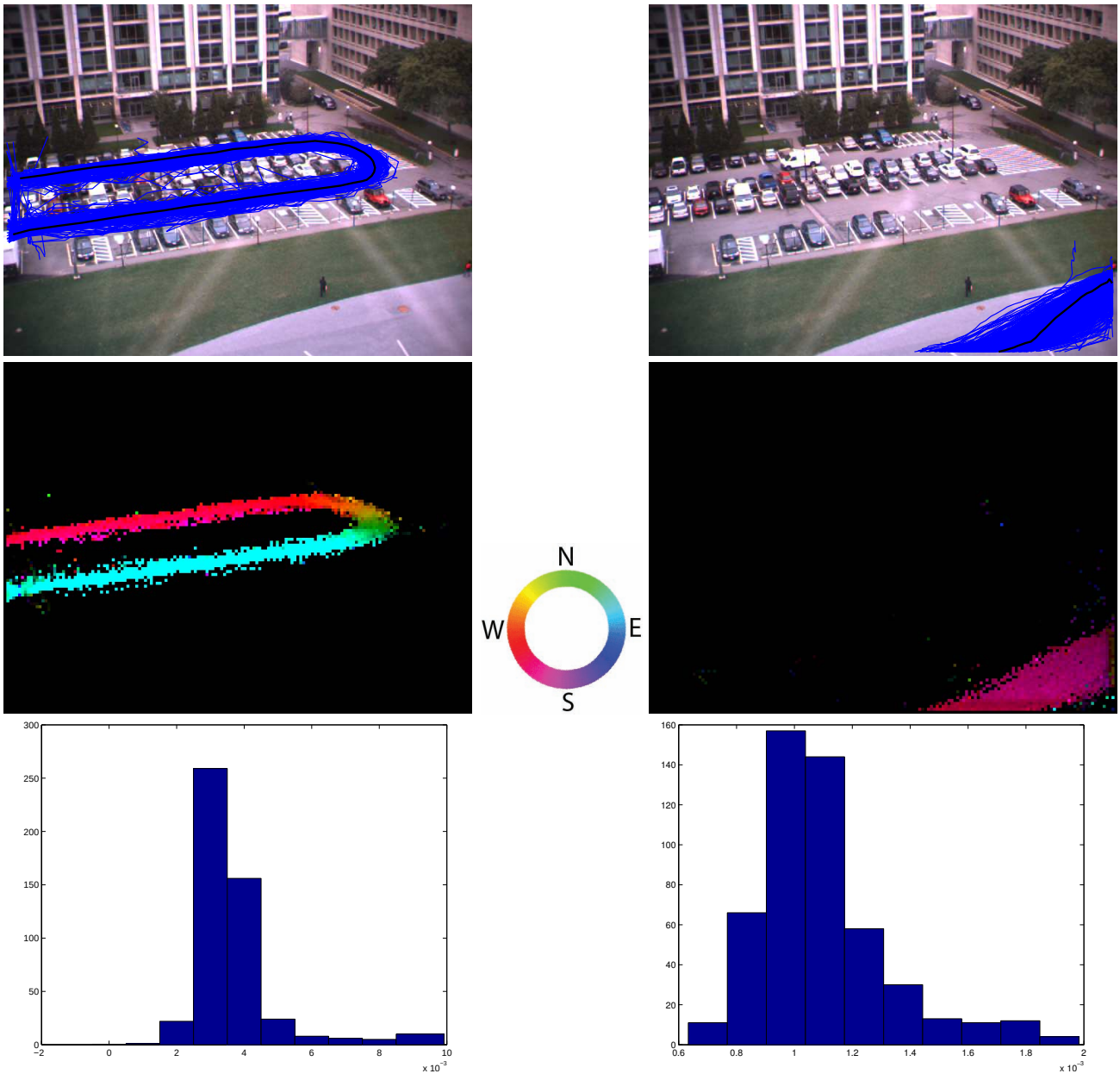


Figure 6-2: A visualization of the top 20 clusters with the highest priors. The most typical track (the one closest to the mean) is shown for each cluster. Color indicates mean size along trajectory (warmer colors correspond to larger size) and arrows indicate the direction of the most typical track.

have a significant effect on the clustering quality—we used $N_E = 20$ for our experiments. Figure 6-1 shows that our clustering quality criterion stabilizes when the number of clusters k reaches 150, suggesting that larger values of k merely result in further subdivision of clusters. Additionally, for values $k \geq 150$, the fraction of inconsistently clustered pairs is very low. Thus we chose $k = 150$ for our experiments.

In any clustering problem, it is valuable to evaluate the clustering quality by inspecting the clusters themselves. Since our goal is to build a scene model, our clusters of tracks should be consistent in terms of the attributes of the cluster members. In the following figures, we show the qualitative results of clustering the set of tracks into 150 groups using position, size, speed and direction of motion as attributes and using the first 20 estimated eigenvectors for the spectral embedding.

Figure 6-2 visualizes the top 20 clusters with the highest priors. For each cluster, we show the most typical track (the one closest to the mean), indicating the mean size of the object along that track with color (warmer colors correspond to larger size), and showing the direction of motion. Note that while pedestrian paths are represented by a cluster in each direction, the car lane in the parking lot (shown in red) is one way.



(a) (b)

Figure 6-3: Two of the most salient clusters in the model. Top row: tracks belonging to each cluster. Middle row: average direction of motion for each grid point shown in HSV space, where the hue is the direction (see the color wheel) and the value is proportional to the magnitude. Bottom row: histogram of average observation sizes.

In Figure 6-3 we examine two of the 20 most salient (in terms of the prior) clusters in the model. The first cluster (6-3(a)) corresponds to cars driving through the parking lot and the second (6-3(b)) represents pedestrian traffic (see the most typical track from each cluster in the top row of Figure 6-6). We overlay the tracks from each cluster on an image of the scene, leaving out the 10% most distant tracks from the cluster mean for better visualization. The most typical track—i.e., the one closest to the cluster mean—is shown in black. Examination of these plots reveals that typical clusters contain tracks that are spatially similar. We further show a visualization of the predominant velocity in the tracks that belong to each cluster. For points on a rectangular grid, we calculate the mean velocity of all tracks in the cluster that pass through the neighborhood of the grid point. We plot the mean velocities in HSV color space, where the hue corresponds to the direction of motion, the saturation is constant and the value (brightness) is proportional to the magnitude of the velocity (speed). Again, we see that the clusters are consistent in terms of the velocity observations. Note that in the car cluster in Figure 6-3(a), the tracks tend to have smaller velocity magnitudes as the cars come around the curve and relatively constant velocities elsewhere. The track of a car that slows down at a different point along the track would not match this cluster well. The outliers in the velocity plots usually correspond to tracking errors. Our clustering is robust to such errors because our distance function averages over the differences in attributes along the entire track. Finally, in the last row of the figure, we show for each cluster the histogram of mean observation sizes in a track (in terms of the bounding box area) over all of the tracks that belong to the cluster. For typical clusters, these histograms tend to have a single peak, suggesting that the member tracks are associated with moving objects of similar size. Note that because some clusters contain tracks in which observations vary greatly in their distance from the camera, the size histograms do not represent the distribution of the true (or normalized) size of the objects. Additionally, at various points along a track, the area of the bounding box may vary as the object moves further or closer to the camera or as it changes pose relative to the camera. This does not pose a problem for our clustering method because we compare spatially corresponding points along tracks. Even though the

size observations will change along the track of a car driving around the parking lot, they should change in a similar fashion in the track of another car following a similar path.

In addition to the two salient clusters in Figure 6-3, we also show visualizations of two clusters with moderately high priors in Figure 6-4—the first contains pedestrian traffic heading away from the building that houses our camera and the second corresponds to people exiting our building and crossing the parking lot to enter another building (see the most typical track from each cluster in the middle row of Figure 6-6). These clusters also show consistency in velocity, position and size.

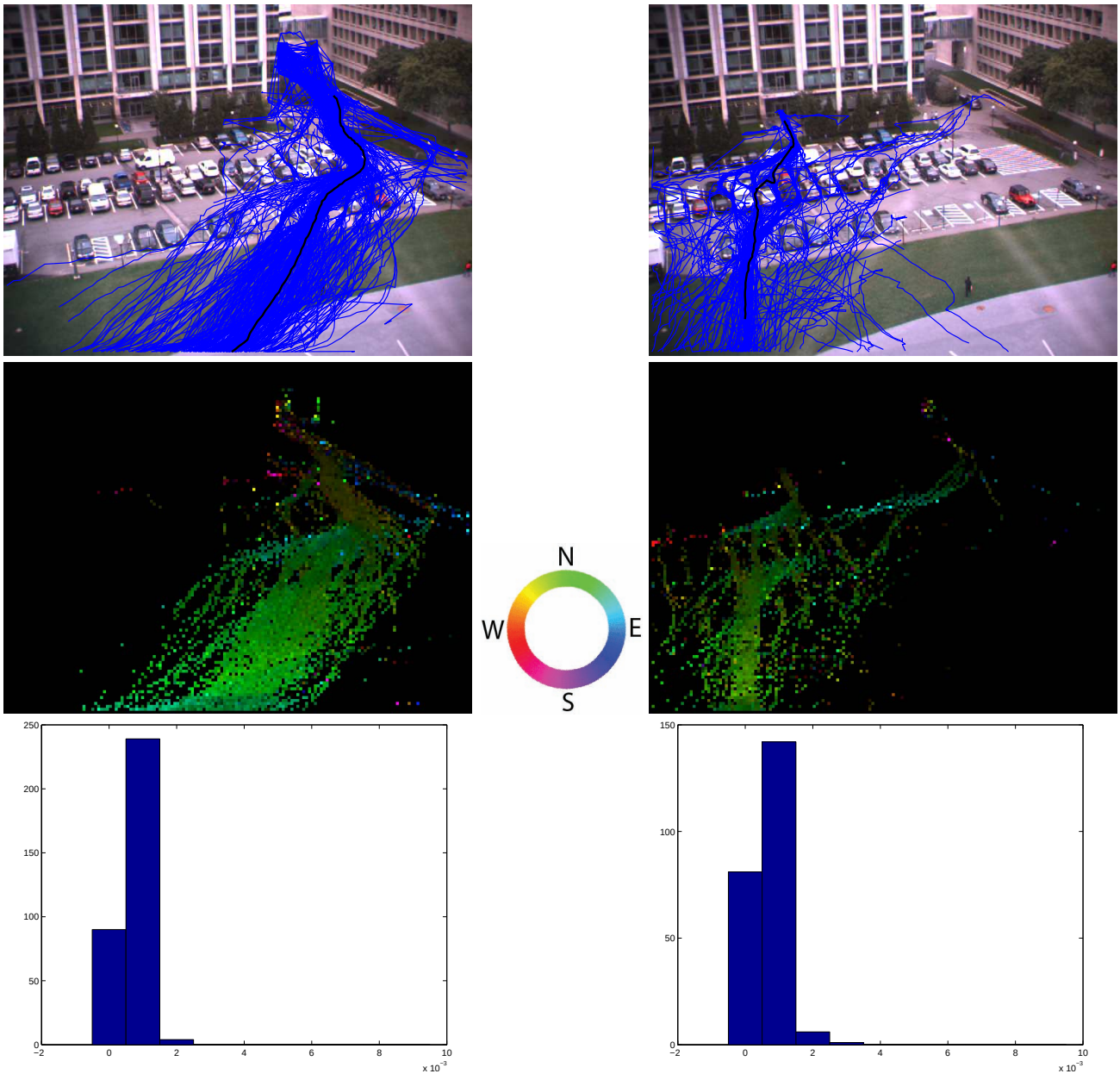
Finally, we show two of the 20 least salient clusters in Figure 6-5. The first (a) corresponds to bicycles—objects larger than the pedestrians in Figure 6-3(b) and moving with greater speed. The second (b) represents cars at night, when our tracking system also tracks the headlight cones, resulting in larger sizes than those in Figure 6-3(a). We show the most typical track from each of these two clusters in the bottom row of Figure 6-6.

In our experiments, the largest cluster has more than 700 examples, while the smallest cluster has approximately 60.

6.3 User Study of Anomaly Perception

One of the goals of activity and scene analysis is to learn what is "normal" and to use that knowledge to detect when something out of the ordinary occurs. The statistical scene modeling approach we have taken in this dissertation makes sense from a mathematical point of view but how does it compare with human perception of anomalous activity?

We conducted a user study to determine whether the likelihood of activity under our learned scene model correlates with how a person familiar with the scene perceives the degree to which the activity is unusual. For instance, if a security guard, whose job was to monitor the scene, flagged certain examples as unusual activity, would those examples also have low likelihood under our model? Conversely, if an example is judged unlikely given our model, would a security guard also flag it as unusual?



(a) (b)

Figure 6-4: Two of the scene model clusters with medium priors. Top row: tracks belonging to each cluster. Middle row: average direction of motion for each grid point shown in HSV space, where the hue is the direction (see the color wheel) and the value is proportional to the magnitude. Bottom row: histogram of average observation sizes.

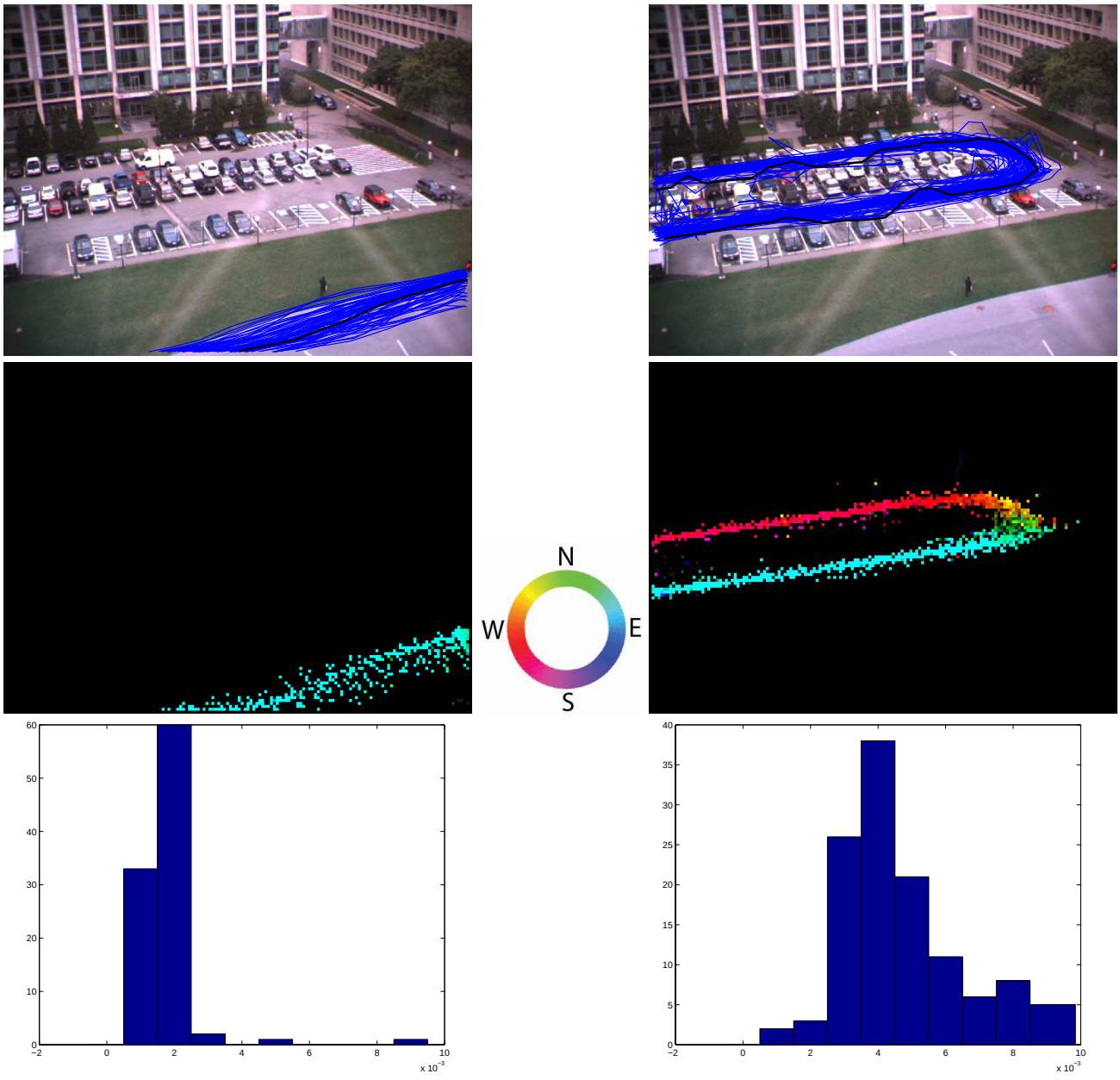
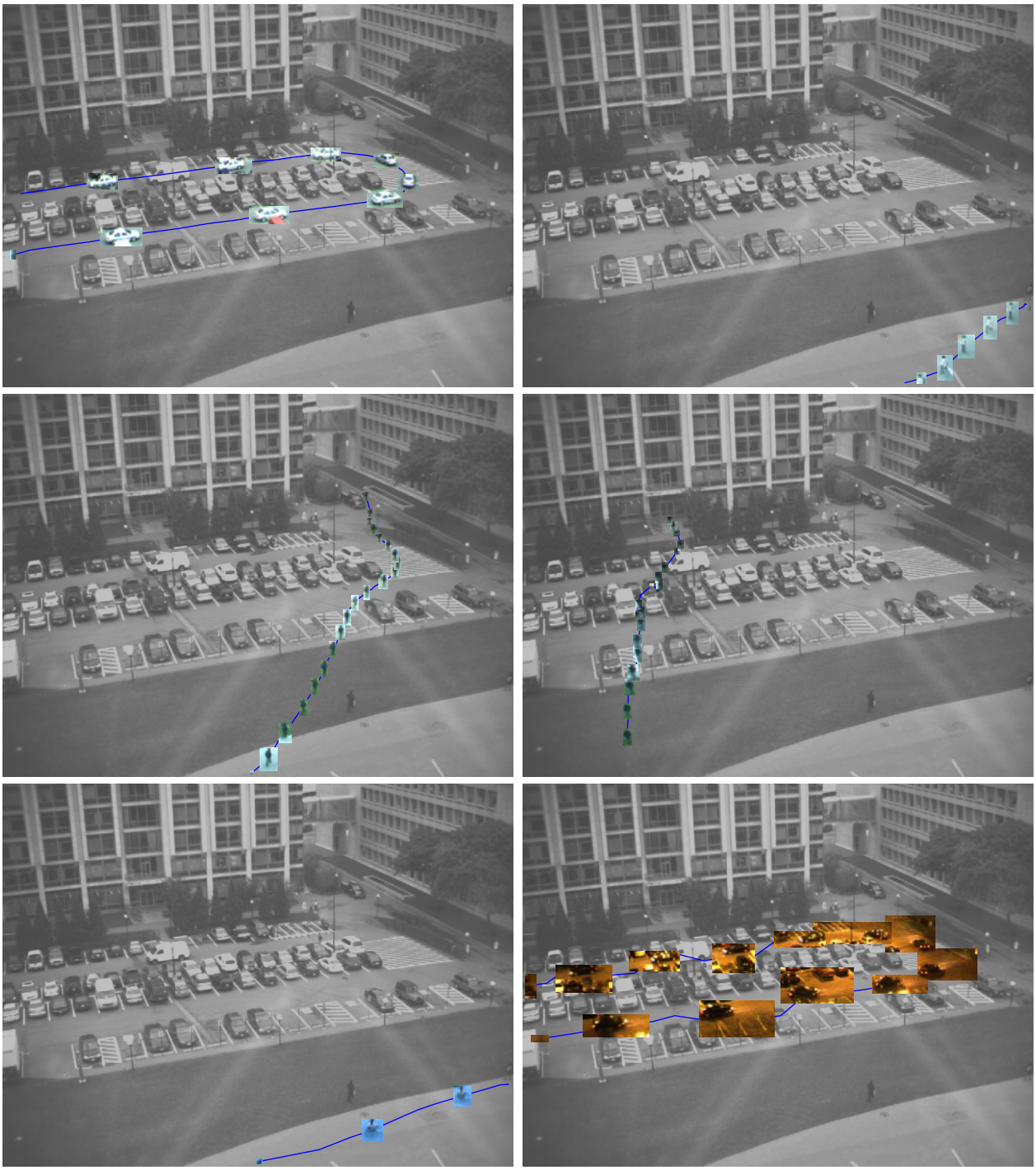


Figure 6-5: Two of the scene model clusters with low priors. Top row: tracks belonging to each cluster. Middle row: average direction of motion for each grid point shown in HSV space, where the hue is the direction (see the color wheel) and the value is proportional to the magnitude. Bottom row: histogram of average observation sizes.



(a) (b)

Figure 6-6: Most typical tracks from the pair of clusters shown in Figure 6-3 (top row), Figure 6-4 (middle row) and Figure 6-5 (bottom row).

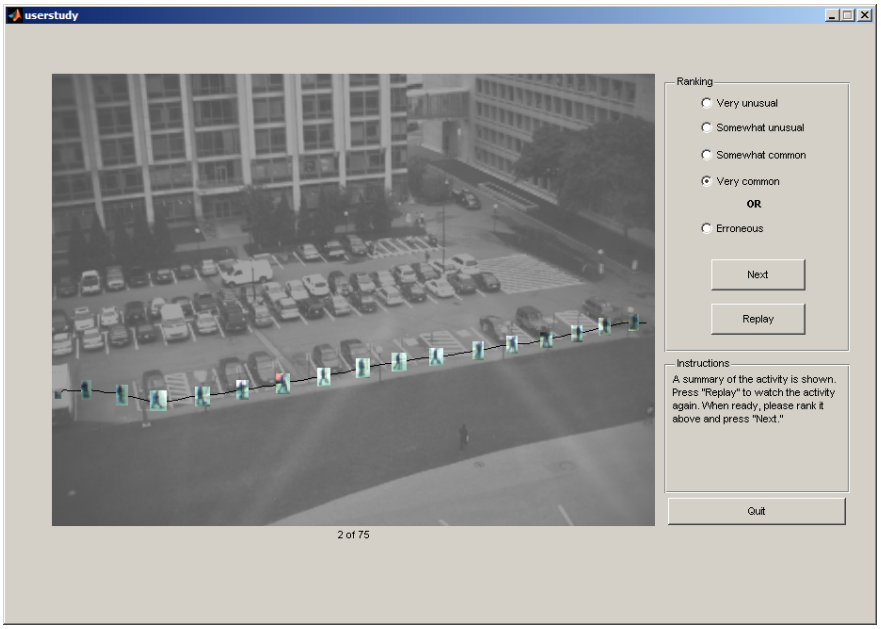


Figure 6-7: The graphical user interface for data collection in our anomaly perception user study. The subjects were presented with examples of activity and asked to rank each according to how unusual it appeared to them given their knowledge of the scene.

6.3.1 Study Design

To approximate the security guard scenario as closely as possible, we recruited experimental subjects who, as residents of our office building, were very familiar with the parking lot scene and could thus be considered experts. We presented each subject with a sequence of 75 examples of activity from the scene. Each activity consisted of the movement of a single object, such as a car, person (or group of people), through the scene, as detected by our tracking system. The examples were presented on a computer monitor as sequences of images, each corresponding to an observation of the object moving through the scene. In each image, the object was shown in the location where it was observed, superimposed on a faded static image of the scene. The pauses between observations corresponded to the time interval between the observations, but to save time the activities were presented at twice the original speed. After the playback of each activity was complete, a summary aggregate image of the activity was shown (see Figure 6-7).

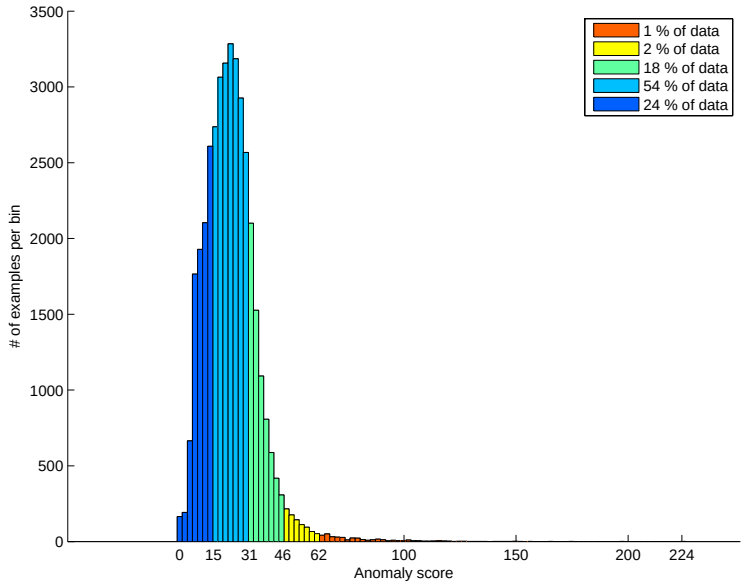


Figure 6-8: Histogram of anomaly scores for the entire dataset (large scores correspond to anomalous examples). Each bar shows the number of examples with the corresponding range of anomaly scores. We partitioned the range of anomaly scores into 5 sections and randomly selected examples from each section for our user study.

We asked the subjects to put themselves in the mindset of a security guard. Their task was to rate each example based on their knowledge of what usually happens in the scene. We asked the subjects to rate (by clicking on the appropriate option with the mouse) each example on the following scale based on how unusual it appeared to them given their knowledge of the location: very common, somewhat common, somewhat unusual, or very unusual. Alternatively, because our object tracking system sometimes made errors, we also gave the subjects the option to label examples as erroneous. Examples of errors included paths of multiple unrelated objects linked together, observations alternating between multiple objects or observations not corresponding to moving objects. The subjects were also made aware that the distribution of examples in the set they would see was not representative of the actual data (i.e. unusual activities may not be rare in the sample).

Because we wanted to correlate the statistical anomaly score (determined from the negative log likelihood of each example under our model) with a human score, we partitioned

the range of anomaly scores (calculated using equation (5.23) and shifted so that the most likely example receives an anomaly score of 0) into five bins, as illustrated in Figure 6-8. We chose the last (most anomalous) bin to contain approximately 1% of the data set and divided the remainder of the range into four bins of equal extent by score. From each bin, we selected 15 examples—75 examples in all—to present to our experimental subjects. Of these 15 examples, 10 were randomly selected without replacement so that each of those examples were ranked by exactly one human. We presented the remaining 5 randomly selected examples from each bin to *all* subjects to obtain a set of 25 examples labeled by all participants. Additionally, before selecting the examples out of each bin, we applied the following rudimentary filters to remove tracks that would not be meaningful to a human observer:

- We removed very short tracks because it would be difficult for human observers to judge whether these are unusual.
- We removed tracks that contained many observations in certain areas of the scene where we know moving objects do not occur. For example, sometimes our system tracks reflections of moving objects in the windows of buildings.

This filtering step (largely due to the removal of short examples) removed about 25% of the examples from each bin.

Our goal in the user study was to assess both how the statistical anomaly scores correlate with human scores as well as how consistent the human scores are. As can be seen in the anomaly score histogram in Figure 6-8, if we had selected a set of examples randomly from the entire dataset without the partioning into bins, the probability of a subject actually ranking examples with high anomaly scores would be small, because the histogram is heavily skewed towards typical examples. Note that the histogram does not have a peak at 0. This is due to the fact that our anomaly scores are proportional to the negative log-likelihoods of examples under a high-dimensional Gaussian mixture. As the dimensionality of a Gaussian cloud increases, because of the sparsity of high-dimensional spaces, proportionally fewer points will occur very close to the mean. We illustrate this point in Figure 6-9 by plotting

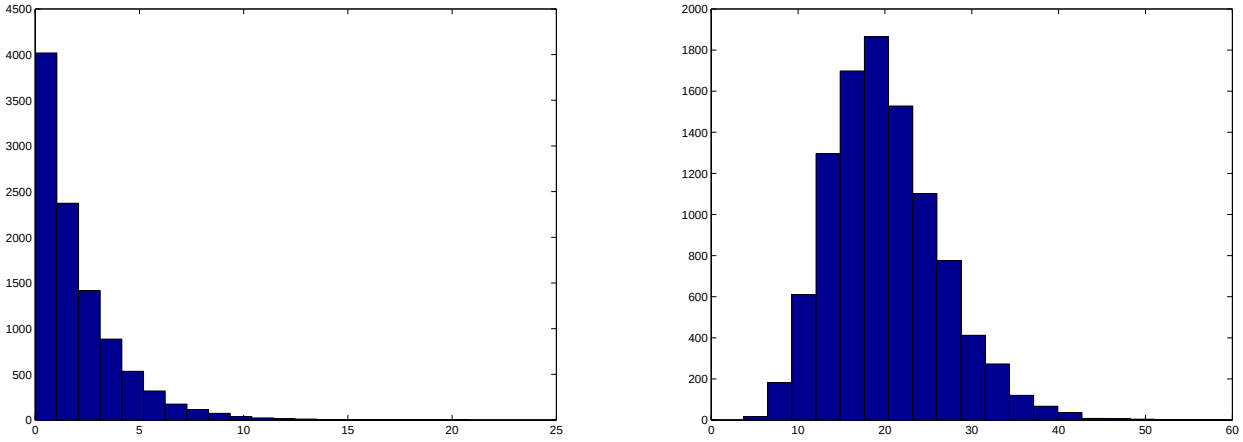


Figure 6-9: The shape of the histogram of squared distances to the centroid for a normal distribution depends on the dimensionality of the space. Histograms of squared distances to the centroid are shown for 10,000 points picked randomly from a standard normal distribution in 2D (a) and 20D (b). In the high dimensional case the histogram's peak is shifted away from 0.

the histograms of squared distances to the origin for 10,000 points drawn randomly from a standard normal distribution in 2D (a) and 20D (b). In the high dimensional case, the histogram's peak is shifted further away from 0. Intuitively, in our track clustering problem, this simply suggests that there will be proportionally more tracks that are very close to the prototype for a given cluster than tracks that are truly prototypical.

6.3.2 Study Results

In this section, we summarize the results of the anomaly perception user study. We have obtained human judgments from 10 subjects, representing 525 labeled examples, of which 25 were labeled by all subjects. Figure 6-10 shows a histogram of human labels for each of the 5 anomaly score bins. For each example labeled by multiple subjects, we take the human label to be the mean label over all subjects. The histogram shows the desired trend: when considering examples from bins with higher anomaly scores, human subjects tend to rank more examples as unusual and fewer as common. Additionally, the proportion of tracking errors increases with the anomaly score, suggesting that on average our system considers

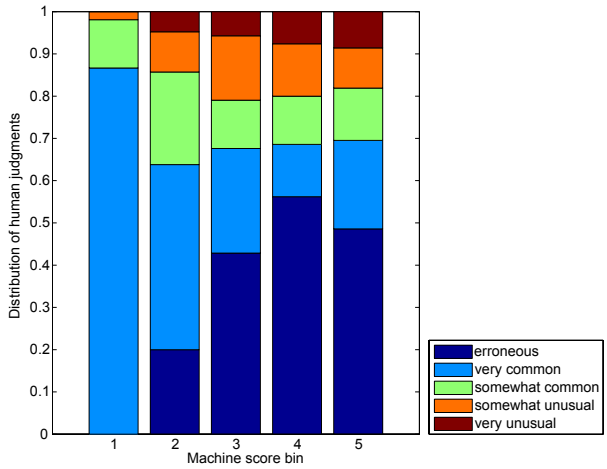


Figure 6-10: The histogram of human judgments for each of the 5 anomaly score bins. The bins are arranged from least to most unusual from left to right.

errors to be unusual. The last (most unusual) bin of the histogram seems to contain more "normal" examples than the previous bin. We believe this is due to the following factors: (1) the last two bins together represent about 3% of the least likely tracks, and it is unlikely that human judgments are precise enough to distinguish these two categories, (2) because of the high error rate in these bins, we have fewer non-erroneous examples from which to estimate the distribution of labels, making the estimates more noisy and (3) the last bin contained some anomalous activities, such as cars driving the wrong way, that were not recognized as such by humans.

Figure 6-11 shows another visualization of the relationship between the statistical anomaly score and the human label. For each of the 5 human labels, we show the anomaly scores of the group of examples that received the corresponding human label. On the left of each column of points we show the mean and variance of the anomaly scores in that group. However, recall that the examples were not picked uniformly from our dataset. Instead, they were picked uniformly from each of the 5 anomaly score bins. Thus to estimate what this figure would look like if we had obtained human judgments for *all* of our examples, we weight those examples that came from heavier bins more than examples that came from lighter bins. The estimated means and variances for an unbiased sample are shown on the

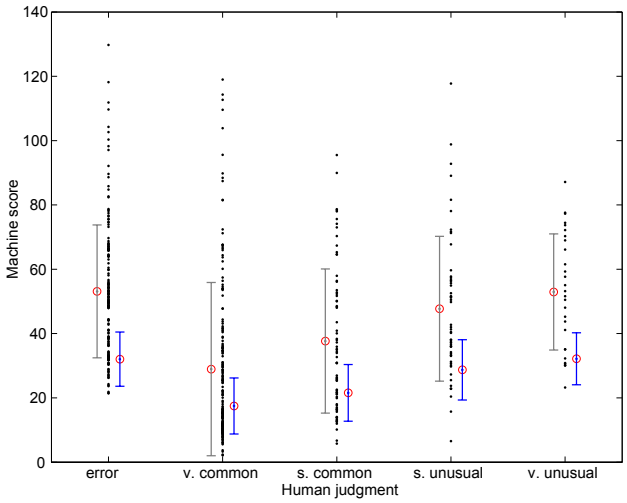


Figure 6-11: The distribution of anomaly scores for each of the 5 different human judgments. The error bar to the left of each column shows the actual mean and variance (for the biased sample), whereas the error bar to the right shows the estimated mean and variance for an unbiased sample.

right side of each group. Again, there is a noticeable trend: examples rated as more unusual by people tend to receive higher anomaly scores by our system. The relatively wide spread of anomaly scores for the different human labels is largely due to two factors: (1) because human responses were influenced by prior knowledge about parking lots in general rather than specific knowledge about *this* location based on long-term observation, some activities recognized by the system as unusual were labeled by humans as common, resulting in false positives; and (2) human subjects were free to use any number of meaningful attributes, such as detailed appearance, as well as inferences about agent intent, resulting in some missed detections. We discuss these cases in more detail in later sections of this chapter.

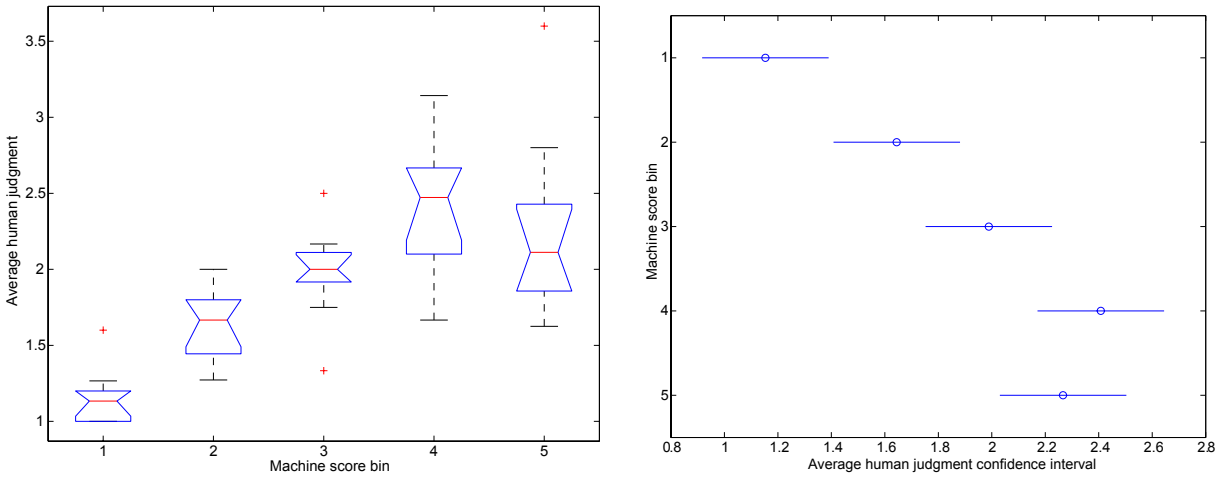
To visualize the statistical significance between these groups of examples, we performed pairwise analysis of variance (ANOVA)[49] for each pair of the 5 anomaly bin groups shown in figure 6-10. To perform this analysis, for each anomaly score bin and each subject, we calculated the average label the subject assigned to examples from that bin, excluding examples marked as erroneous. This results in the matrix shown in Table 6.1. Most of the subjects tended to rate examples from more unlikely anomaly score bins as more unusual. At

Subject #	Bin 1	Bin 2	Bin 3	Bin 4	Bin 5
1	1.20	1.89	2.50	2.67	3.60
2	1.27	1.50	2.17	2.55	1.90
3	1.00	1.44	2.11	2.33	2.43
4	1.60	1.80	2.11	2.67	2.80
5	1.13	2.00	1.75	3.14	1.86
6	1.13	1.67	2.00	2.56	2.40
7	1.00	1.43	2.00	2.10	1.92
8	1.00	1.78	1.92	2.00	1.63
9	1.13	1.67	2.00	2.40	2.30
10	1.07	1.27	1.33	1.67	1.83

Table 6.1: A table showing each subject's average label assigned to examples from each anomaly score bin, excluding examples marked as erroneous. Numerical values of the labels are as follows: very common (1), somewhat common (2), somewhat unusual (3), very unusual (4).

first examination, the data for the last, least likely bin seems to contradict this trend: several of the subjects gave a lower average rank to examples from this bin than to examples from much more common bins. Examination of examples that came from this bin but were labeled as very common revealed several instances of cars traveling the wrong way in the parking lot. Though some of our subjects consistently marked such examples as very unusual, knowing that the parking lot is one way, other subjects failed to realize this and marked similar examples as very common. Because the rate of tracking errors in this bin was high, even a single example marked as very common lowers the subject's average label significantly.

A common way of examining groups of labels like those in table 6.1 is by visualizing them as box plots. Figure 6-12(a) shows a box plot, in which each group represents a column of the matrix in table 6.1. The red bars indicate the medians, while the boxes mark the upper and lower quartiles. The whiskers show the range of the data in the group. The figure again shows the desired nearly monotonic relationship that we observed in Figure 6-11. Part (b) of Figure 6-12 shows the analysis of variance comparisons between the groups from part (a), i.e. the columns of the matrix in table 6.1. Each pair of groups for which the bars in the plot do not overlap have statistically significantly different means with 95% confidence ($p \leq .05$). We can see that even with our limited number of experimental subjects, each of bins 1 and



(a) (b)

Figure 6-12: (a) Box plot of the mean human judgments (one data point per subject) for each of the 5 anomaly score bins. The bins are arranged from least to most unusual from left to right. The red bars indicate the medians, the boxes show the upper and lower quartile ranges and the whiskers show the range of the data. Pluses (+) indicate outliers. (b) Pairwise ANOVA on each group from (a). Circles show the mean values for each group and bars denote the 95% confidence range for the mean (means with non-overlapping bars are statistically significantly different with 95% confidence).

2 (the most common bins) received significantly different human judgments from bins 4 and 5 (the most unusual bins).

Since we have obtained a machine score (the anomaly score) and a human score for each example in our user study, we can also directly calculate how well these two scores are correlated. However, because both scores are on different scales and the relationship may not be linear, we used Spearman's rank correlation coefficient [51] (often called Spearman's Rho) as a measure of the correlation between the two scores. Spearman's Rho operates on the statistical rank, i.e. the ordinal number of each score value when ordered monotonically, rather than on the scores itself. Thus it is considered to be nonparametric, in contrast with the correlation coefficient, which assumes a linear relationship between the scores. The definition of Spearman's Rho is as follows:

$$\rho = 1 - 6 \sum \frac{d^2}{N(N^2 - 1)}, \tag{6.2}$$

where d is the difference between the statistical rank of corresponding scores and N is the number of samples. For our user study, the rank correlation coefficient between the statistical anomaly scores and the human labels for examples not labeled as errors was $\rho = 0.43$ with a p-value $p < e^{-10}$. The p-value [50] in this case is the probability of a rank correlation larger than or equal to ρ occuring by chance.

To compare this result with how well humans are able to predict each other's responses, we can calculate the statistical rank correlation between each set of human labels and the *average* human label on the set of 25 examples that were seen by all of our subjects. As before, in each comparison we exclude examples labeled by the person as an error. We find that for those subjects for which the Spearman's Rho between their labels and the average human label has a p-value $p < .05$, the worst correlation found was $\rho = 0.61$ and the best was $\rho = 0.90$. Thus the human labels are a better predictor for the average human ranking than our statistical anomaly score, but our score bears a strong relationship to the human labels as well. This result is not unexpected, as our human subjects, though familiar with the scene, have not spent significant amounts of time watching the scene. It is reasonable

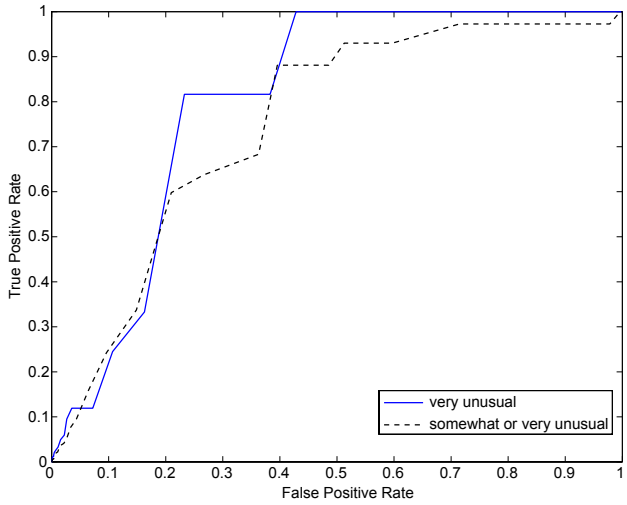


Figure 6-13: The estimated ROC curve for the detection of unusual events (dashed line) and very unusual events (solid line). Each point on the curve corresponds to a different anomaly score threshold for positive detections. False positives are examples that fall above this threshold but were classified as less than very unusual by humans (solid line) or less than somewhat unusual (dashed line).

to assume that their judgments are heavily influenced by their prior beliefs of what usually happens in parking lots in general rather than what happens in *this* parking lot.

Another way of thinking about our track-based scene modeling framework is in the context of a classifier. Suppose we wanted to use our model to classify activity examples into two groups: "unusual" and "not unusual". To do this, we would simply set a threshold on the statistical anomaly score and label as unusual each activity that receives a higher score than the threshold. Taking advantage of the fact that we have a set of examples labeled by humans, we can attempt to estimate an ROC curve for such a binary classifier. We can define true positives as those examples that have anomaly scores higher than the threshold and were labeled by humans as "very unusual." Coversely, false positives are examples for which the anomaly score exceeds the threshold, but the human label is something other than "very unusual." This time, we include the errors into the analysis. For various values of the threshold, we can now calculate the ratio of the true positive rate to the false positive rate. However, again we have to address the fact that our sample was not drawn randomly. Thus,

we weight each example by the proportional weight of the bin from which it came. This is akin to repeating examples from "heavier" bins. The resulting ROC curve is shown in Figure 6-13. We also show the ROC curve when true positives are defined as examples that received anomaly scores above the threshold and were labeled as humans as "very unusual" or "somewhat unusual." The area under the curve is approximately 0.81 for the detection of very unusual events and 0.75 for the detection of unusual events (chance would be 0.5), suggesting that our anomaly scores would lead to a reasonable classifier for unusual activity.

Note that to estimate the ROC curve, we essentially considered the human judgments obtained through our user study to be the ground truth. In reality, the performance of our algorithm may be better than it appears, as the ground truth labels are probably heavily influenced by our subjects' prior beliefs about parking lots in general. Nevertheless, even considering our human labels to be the ground truth, if we chose the anomaly score threshold to be such that we detect at least 80% of the examples labeled as "very unusual," we would be filtering out approximately 80% of the data and our false positive rate would be about 23%. This is a reasonable level of performance, especially considering the fact that the human labeling task was fairly unconstrained and the human judgments were certainly not limited to only the attributes we used to learn the model.

6.4 Detected Activity: True and False Positives

In this section, we take a look at examples of unusual activities detected using our system. All of the examples shown here came from the two bins representing the most unusual 3% of the dataset. In Figures 6-14 and 6-15, we show those detections that were also ranked as "very unusual" by our human subjects. In Figure 6-16, we show examples that may be considered the most illuminating false positives in the sense that they were among the most unusual 3% of the dataset, yet they were labeled as "somewhat common" or "common" by human subjects.

Consider first the true positive detections in Figure 6-14. For each detection, we show a summary image of the activity, in which we plot the trajectory of the moving object and

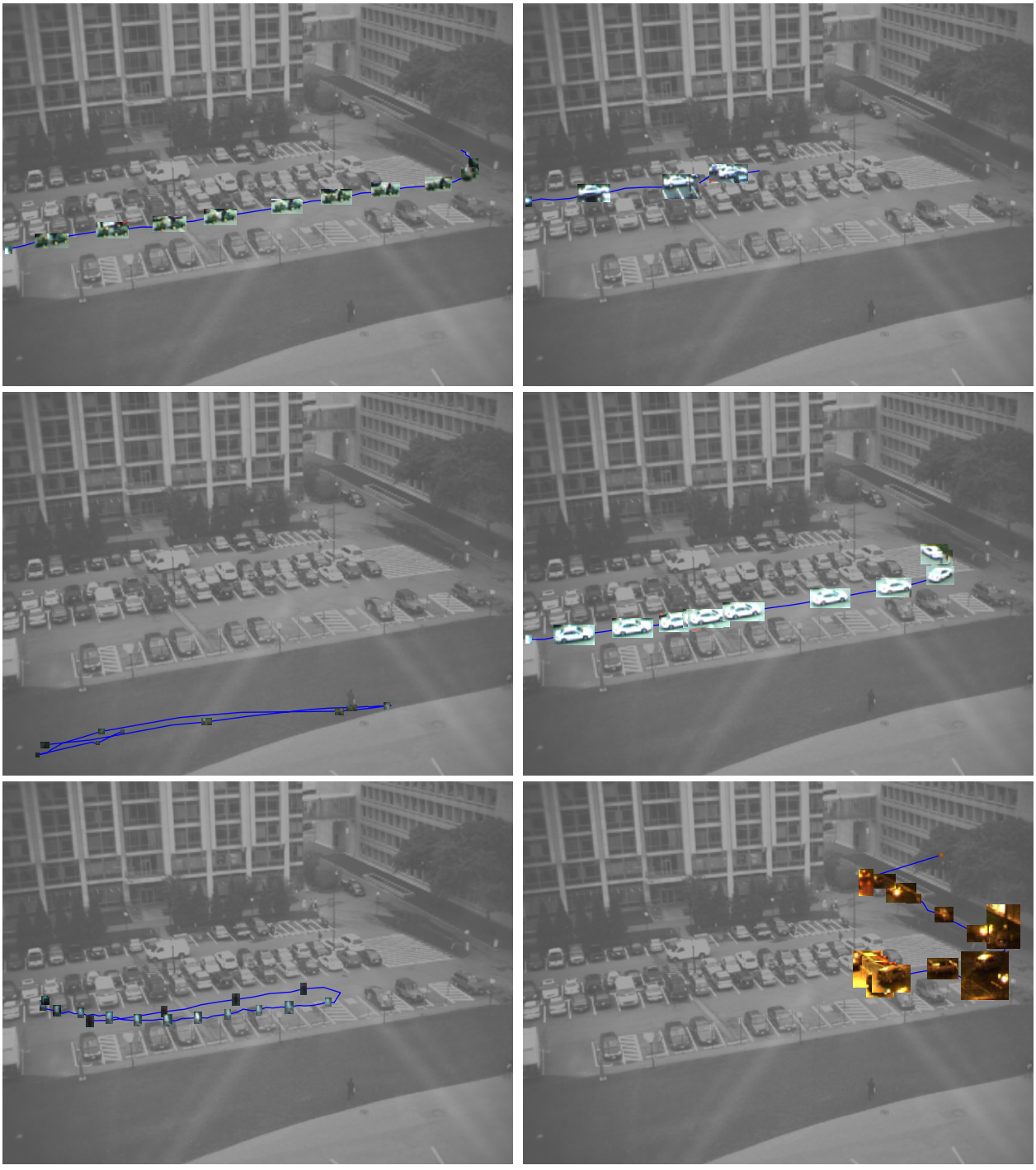


Figure 6-14: True positive detections: examples that received anomaly scores in the top 3% of the data and were also labeled by humans as "very unusual."

superimpose on a faded image of the scene the contents of the bounding box of observations approximately 2 seconds apart. Thus the more sparse the observations, the faster the object was moving. Following is an annotation of Figure 6-14 in counter-clockwise order:

1. A small landscaping tractor pulling a trailer drives through the parking lot and enters the pedestrian area. For a large object, both the path (entering the pavement) and the speed (moving slower than cars) are unusual.

2. A dog runs back and forth on the grassy area (possibly playing fetch). Again, both the path and speed of motion are unusual for a track in this area of the scene.

3. An object that moves fast through the parking lot, then changes direction and moves more slowly. This is an example of a track that was labeled as very unusual but plausible (not erroneus) by a human and very anomalous by our system, yet may actually be a tracking error. Such examples demonstrate the need for an attentive system that could task a higher resolution camera to collect images that could be used to ascertain whether this was truly an unusual activity.

4. A car leaving a parking spot and driving away through the pedestrian area of the parking lot (these observations have a different appearance because they took place at night).

5. A car drives through the parking lot, slows down to a halt, then continues to drive to a further spot, where it stops again for a significant period of time. Note that in this case the anomaly is temporal in nature: the *sequence* of the velocities in the track is what is unusual, rather than the size or the trajectory. Approaches that do not consider multiple attributes of tracks in a principled manner might have a difficult time detecting such activities.

6. A car drives through the parking lot in the wrong direction. This track is unusual because objects of this size taking similar paths through the parking lot usually move in the opposite direction.

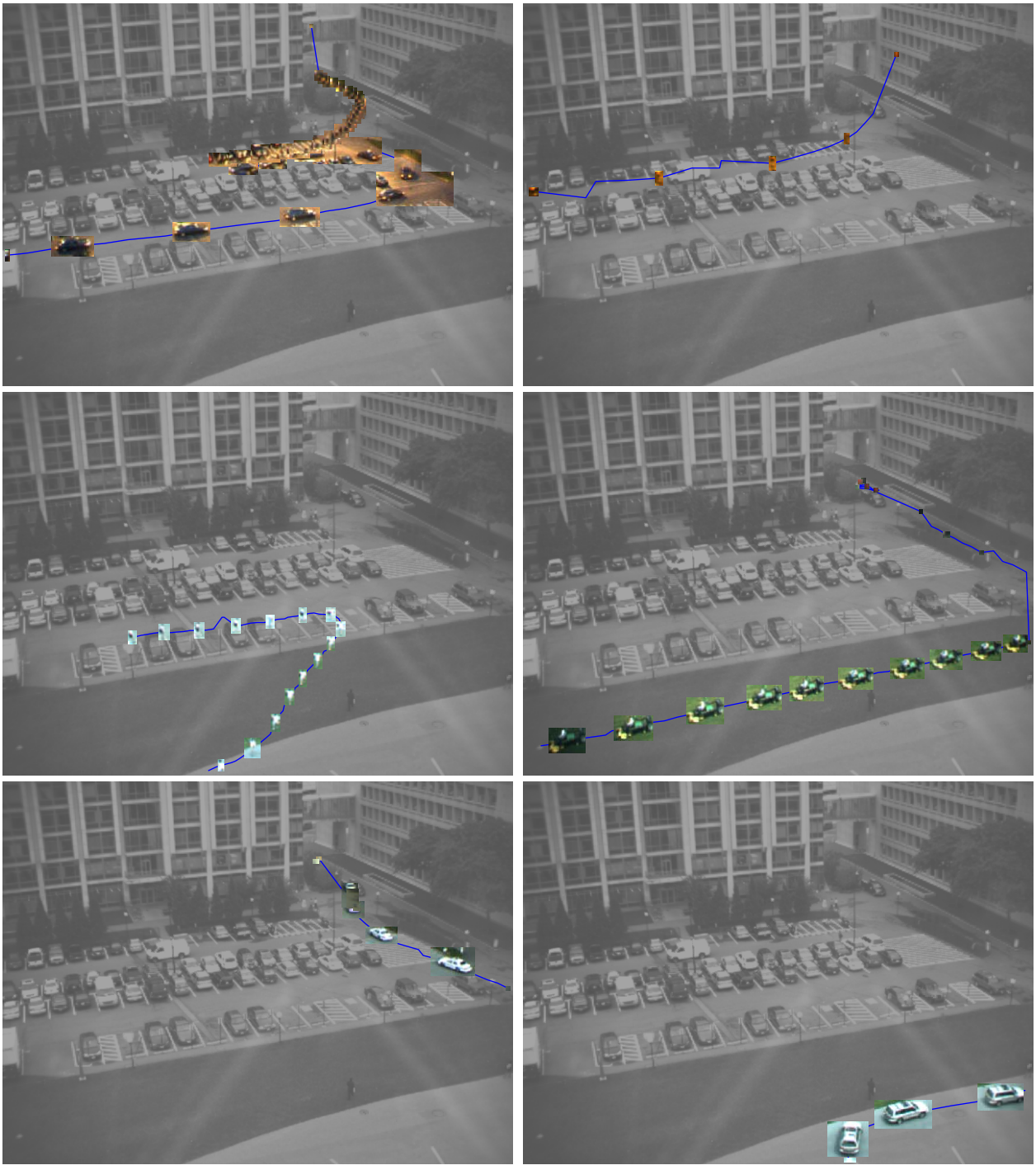


Figure 6-15: True positive detections: examples that received anomaly scores in the top 3% of the data and were also labeled by humans as very unusual.

The annotation for Figure 6-15, again in counter-clockwise order, is as follows:

1. A car drops off a passenger in an unusual spot in the parking lot. The passenger then walks away towards a distant building. In this case the trajectory is unusual.

2. A person walking through the parking lot suddenly changes direction. Again, the path of the motion is unusual.

3. A police car cuts across the parking lot, coming from a pedestrian area and moving into another pedestrian area. This activity would be normal for a small object, but is unusual for a car.

4. A van drives through a pedestrian zone. Again, the sizes of the observations are unusual in this area of the scene, even though the trajectory is common.

5. A lawn mower is detected on the grassy area. Objects of such size moving perpendicularly to the pavement are rare.

6. A small object, probably a person on a skateboard, moves very fast through the parking lot. The speed of the motion is unusual for an object taking this path through the scene.

Let us now take a look at the false positives: examples that received anomaly scores in the top 3% of the data yet were labeled by our subjects as common. Following is the annotation of Figure 6-16 in counter-clockwise order:

1. A large group of people moves through the pedestrian area. Though pedestrians and even smaller groups are very common, large groups of this size are more rare. However, a human observer would probably conclude, based on his prior knowledge of similar scenes in general, that such an activity is typical.

2. Another example of several people traveling together through the scene. This is another example of an activity that may not be common, yet might be perceived as such by a human observer.

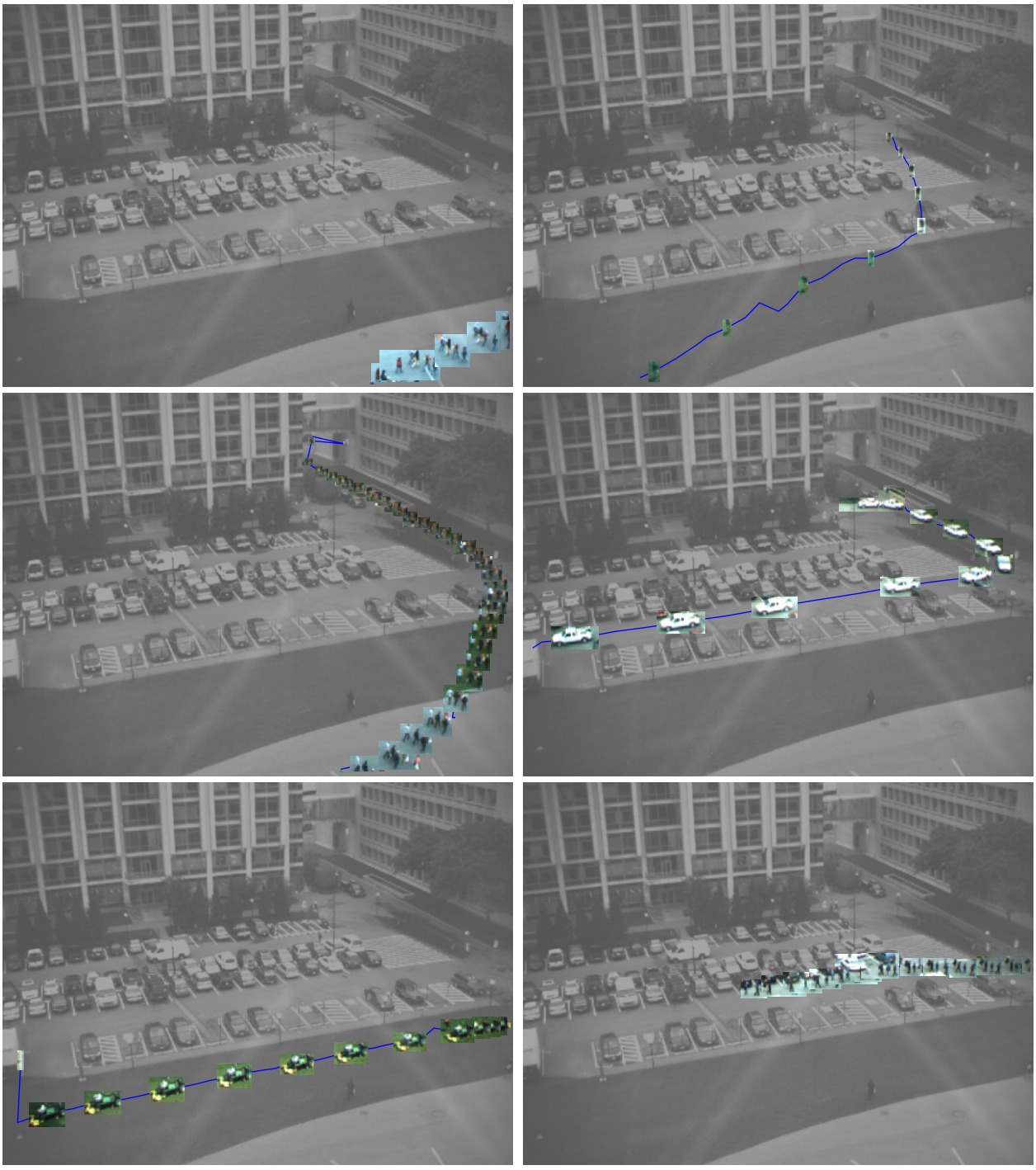


Figure 6-16: False positive detections: examples that received anomaly scores in the top 3% of the data and were labeled by humans as common (but not as erroneous).

3. An example of an activity that happens infrequently (a lawnmower on the grassy region) and so received a high anomaly score, yet may be perceived as common or unusual, depending on the observer's mindset. A very similar example occured in our true positives: a human thought it was unusual, probably because it is rare. However, a different person might label this activity as common, because such a path through the scene *is* common for a lawn mower. If we considered the appearance of the object as an attribute and collected data over a much longer period of time, our model might have a cluster of lawn mowers taking similar paths through the scene.

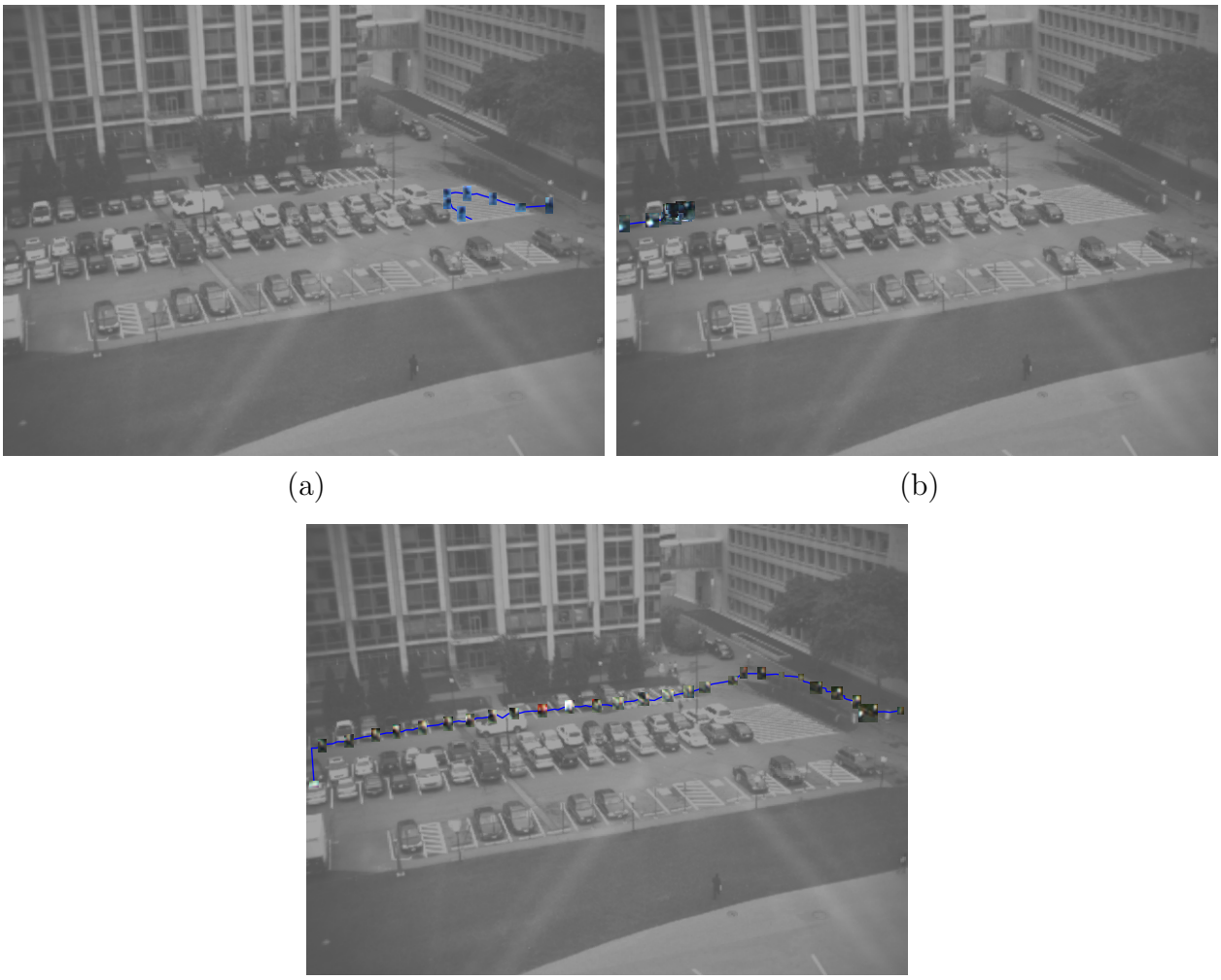
4. Another example of a large group of people moving through the parking lot.

5. A car taking an unusual path through the scene. It is not clear why a person labeled this as common, unless he or she did not realize that the car was driving through a pedestrian area and left the one-way parking lot in the wrong direction.

6. A person running through the scene. Though runners are much more rare than walkers in this scene, an observer—even one familiar with the scene—might not realize it.

6.5 Missed Detections

In the previous section, we looked at examples from the least likely 3% of the data set. As was clear from our ROC curve in Figure 6-13, not all activities labeled as very unusual by our subjects received such high anomaly scores. Figure 6-17 shows two typical examples of missed detections: examples that received anomaly scores in the two most likely bins, yet were labeled by humans as very unusual. We show these examples because they reveal possible failure modes of our approach. In the first example (a), a person took an unusual path through the scene, but the segment of the track after the person changed directions is very small. In this case, a human observer can tell that something unusual happened but our distance function may not reveal it unless the track were longer or we used other attributes such as a second derivative of the velocity. In the second example (b), it can be seen from a



(a) (b)

(c)

Figure 6-17: Missed detections: examples that have anomaly scores in the two most common bins yet were labeled by humans as very unusual.

movie of the sequence of observations that a person approached a car, opened the hood and spent some time working on something inside. It would be difficult to recognize this activity as unusual without considering the appearance of the observations. In the final example (c), the track was probably perceived as unusual because the person took did not take the shorter route, opting instead to walk around the other side of the grassy area. In this case, we would have to take into account semantic information about the scene in order to recognize this activity as unusual.

These three types of missed detections contribute to the relatively wide spread of statistical anomaly scores for examples labeled as very unusual (see Figure 6-11)—unlike our system, which uses a limited number of descriptive attributes—the human subjects were free to use additional attributes, such as detailed appearance and inferences about the intent of the agents involved in the activities.

6.6 Clustering with Time of Day

In addition to the clustering experiment used for the user study, we also experimented with using time of day as an additional attribute. For a track T in our data set, the time of day is the time of the first observation. We normalize the time difference for track comparisons such that a value of 0 indicates identical time of day for both tracks and a value of 1 indicates a maximum (i.e. 12 hour) time difference. Using position, velocity, size and time of day as attributes, we clustered the data set into 250 clusters. The cluster number was chosen in a similar manner as in the previous experiment, by evaluating the quality of the clustering.

In Figure 6-18, we show examples of activities that were in the two least likely bins of the data set in terms of the statistical anomaly score when time of day was considered, but in the two most likely bins if time of day was *not* considered. Thus these are examples of activities that would be considered normal if we did not know at what time they occured. In all of the shown examples, we print the time of day in the top left corner of the image. All of these moving objects took normal paths through the scene but at unusual times during the day.

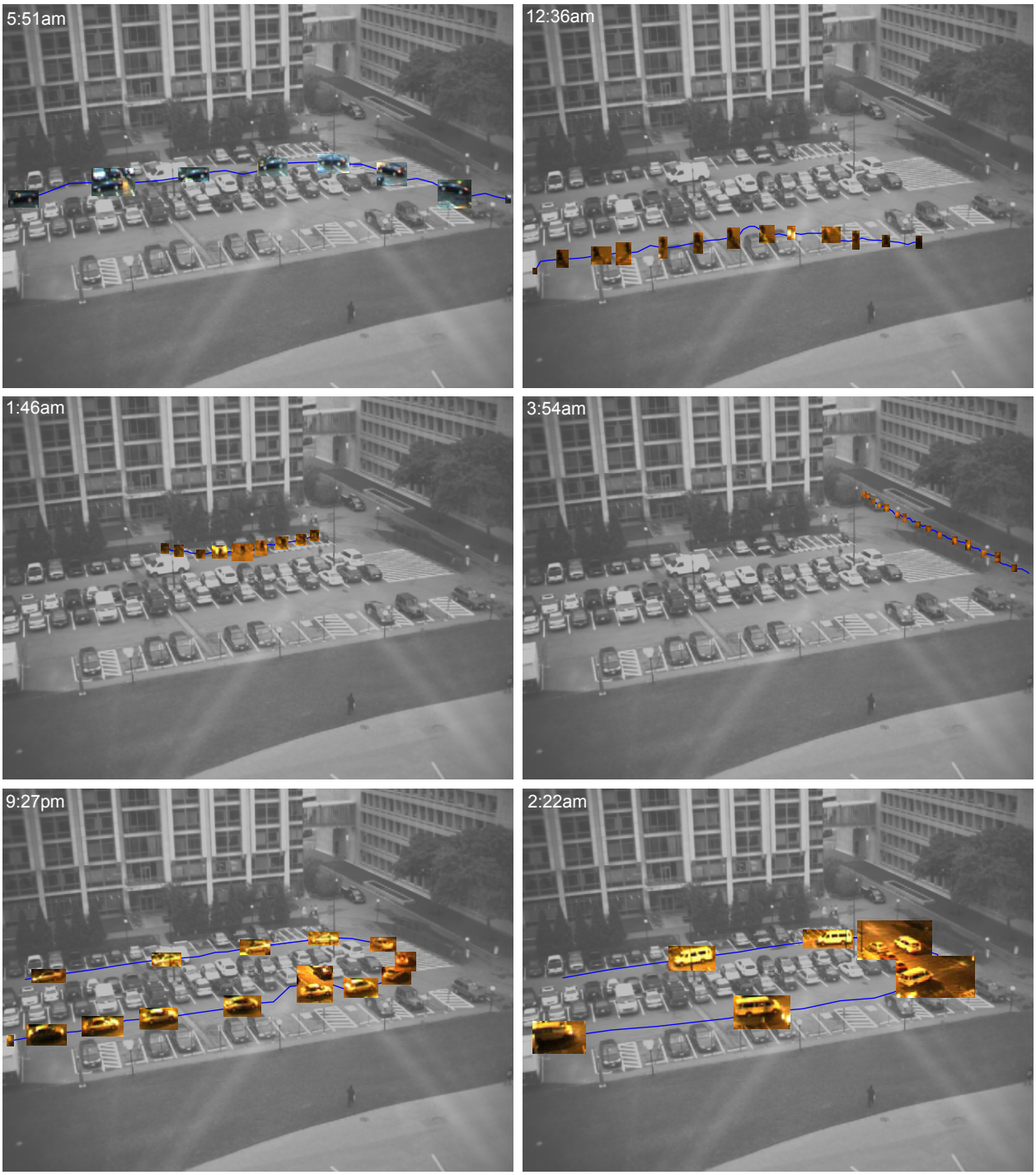


Figure 6-18: Examples of statistically anomalous activity when time of day is considered as an additional attribute. All shown examples appear normal when time of day is not included in the comparisons between tracks. The time of day for each example is shown in the upper left corner.

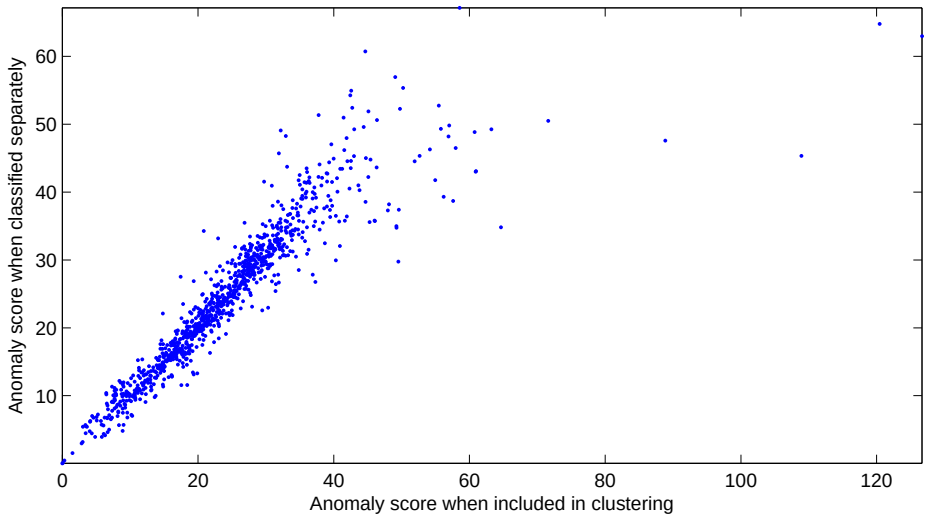


Figure 6-19: Anomaly scores for 1000 randomly selected examples. The x-axis corresponds to the anomaly scores when clustered together with the rest of the data set, whereas the y-axis shows the corresponding anomaly scores when left out for the learning phase and then classified by assignment to the best matching cluster. The two scores are very well correlated.

6.7 Stability of Anomaly Detection

To assess the stability of our anomaly scoring method, we performed the following experiment. We randomly selected 1,000 tracks from the dataset (excluding the Nyström sample) and set these examples aside as a testing set. We then learned a scene model using the same parameters as before from the remaining learning set. Finally, we classified each of the 1,000 examples in the test set using the classification method outlined in section 5.6 and we calculated their anomaly scores under the model. If our anomaly scoring method is stable, the anomaly scores of the examples from the test set should be strongly correlated to the anomaly scores of those same examples when they were included in the learning set. Figure 6-19 confirms this. The two sets of anomaly scores are very strongly correlated, suggesting that our algorithm would perform well on new examples of activity not included in the learning set.

6.8 Surprise Detection

In section 5.9, we described a method for the detection of surprising moments along developing moving object tracks. For a developing track, we calculate at regular intervals the posterior distribution over the clusters (mixture components) in our model. Whenever there is a large change in this distribution as measured by the KL divergence (relative entropy), the corresponding observation marks a surprising moment in the track.

We show experimental results of this type of surprise detection in Figure 6-20. For four different activities (tracks), we show a summary image in which the observations are numbered. For each track, we calculate the posterior distribution over the mixture components at regular intervals (2 seconds) as the track develops, starting 2 seconds into the track. We plot the KL divergence between the distribution at the current time and the previous time (2 seconds ago). Peaks in the KL divergence plot should correspond to surprising moments in the track. Consider the four activities in Figure 6-20, counter-clockwise:

1. A person walks through the pedestrian zone, suddenly changes direction and walks back. The surprise (a peak in the KL divergence plot) occurs around observation 28, after the person has just turned around. Intuitively, this means that the observations immediately after the pedestrian turned about are surprising, given the previous observations in the track.

2. An example of an unusual activity that does not have a distinct surprising moment. The KL divergence plot is relatively flat. It is high at the first point where relative entropy was calculated because a few observations of a large object, e.g. a group of people, in this area are not necessarily rare, but a large object moving parallel to the pavement is unusual.

3. Another example of an activity that does not have a surprising moment. This time it is a common activity.

4. An activity with several surprising moments. A cyclist bikes through the scene, but does not enter the pedestrian area around observation 28, as would be usual. Instead,

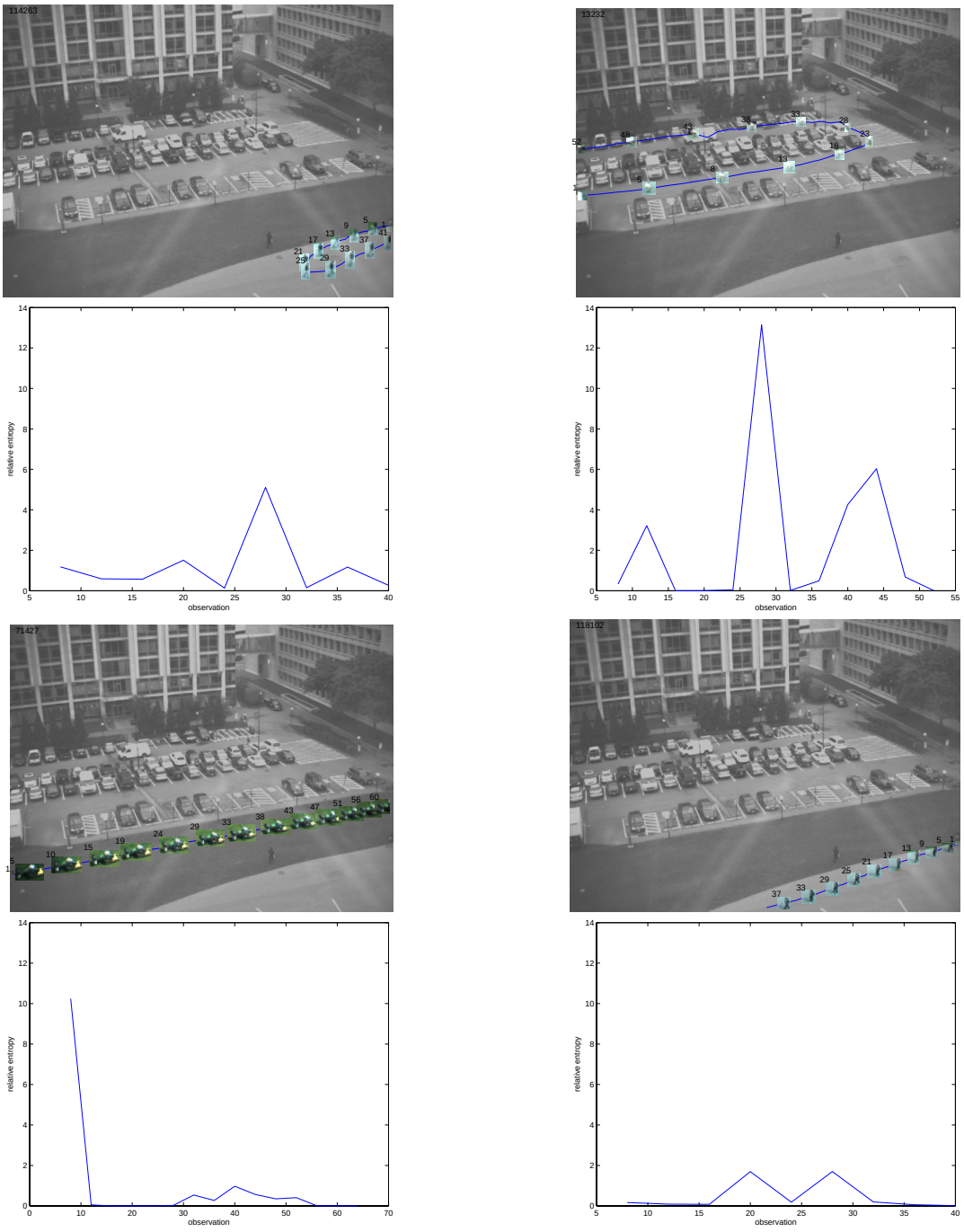


Figure 6-20: Surprise detection experiments. Each image shows an activity with a subset of the observations labeled. The corresponding plot beneath each image shows changes in the belief distribution over the clusters in the scene model. For each obseration, we plot the relative entropy between the current and the previous belief distribution. Surprising moments along the tracks should correspond to large changes in the distribution.

he turns around and bikes back through the parking lot. There is another surprising moment when the cyclist does not turn towards the bike rack in front of the office building.

These early experiments with track-level surprise detection show promising results. The surprise detection framework could be used in conjunction with the statistical anomaly detection to task a high resolution camera to interesting or surprising activities.

6.9 Summary

In this chapter, we evaluated our track-based attention model both qualitatively, by looking at individual clusters, and quantitatively, by comparing our statistical anomaly scores with human judgments. The results show that the statistical anomaly score correlates well with human perception of anomalous activities, and thus our model could realistically be used in conjunction with an attentive monitoring system. In the following chapter, we conclude the dissertation with a discussion of some of the remaining open questions.

Chapter 7

Conclusions

The last decade has seen an incredible increase in the availability of sensors, network bandwidth and computer processing power. As a result, the amount of information potentially available to a modern intelligent monitoring system is simply too vast to process in its entirety. One way to address this issue is by developing attentive mechanisms that would allow monitoring systems to recognize parts of the input that are in some way more interesting than the rest. The goal of this dissertation was to apply this concept to the visual surveillance domain. We have demonstrated a data-driven attentive mechanism that makes it possible to learn a model of activity in a far-field scene and apply the model to the problem of determining how unusual a new activity is.

7.1 Contributions

Our work contributes to the field of scene and activity analysis in several important aspects:

- We evaluate the performance of our scene model by comparing the statistical anomaly scores with human perceptual judgments on a set of examples labeled by experimental subjects. This enables us to estimate how well our system would perform if it were used as a classifier for activities that people would perceive as unusual. To our knowledge, no other statistical scene modeling approach has been evaluated in this fashion. We show

that our anomaly scores are well correlated with human judgments, suggesting that our methods could be used in practice to support attentive mechanisms in automatic monitoring systems.

- Our scene model contains a rich description of the usual pattern of object motion in the scene due to the fact that we incorporate multiple object attributes. Our experiments demonstrate that we can detect many types of unusual activities that would be difficult to spot using an approach that did not combine multiple attributes in a principled manner.

- We show results on a collection of activities an order of magnitude larger than can be found in the related published work. This serves two important purposes: (1) we can build a more descriptive model (with a large number of clusters) without the danger of overfitting and (2) it enables us to experiment with a sizeable set of rare events, which would ordinarily only occur a handful of times (if at all) in much smaller datasets.

7.2 Applications

We envision several possible applications of our work with potentially great impact on the visual monitoring community.

In most modern monitoring systems, a human operator ultimately decides whether a detected activity warrants further investigation. This is partly due to the fact that artificial intelligence algorithms are not yet powerful enough to automatically recognize certain types of activity, such as public safety risks or suspicious behavior. The methods from this dissertation could be applied to filter out activity that is common based on long-term observation of the scene. In particular, as we showed in section 6.3.2, at the threshold for which 80% of examples labeled by people as very unusual are detected, our system filters out about 80% of the data. Of the remaining 20% of the data, about 23% of the detections would be false positives. As a result, a single operator would be able to oversee a much larger network of sensors than has previously been possible. Based on conversations with industry

professionals, a relatively high rate of false positives is acceptable in return for a high *true* positive rate. Further, in systems that have a movable high-resolution camera, our methods could be used to automatically task that camera to obtain detailed images of an activity recognized as unusual while it is in progress.

Another application is in data retrieval and classification. Having built a statistical model of activity in the scene, we can use the model not only to recognize unusual activity, but also to find *other* activities that are similar, i.e. close to each other in the spectral embedding space. For instance, given a particular example of activity, we can retreive other examples that have been classified as belonging to the same cluster (mixture component in our scene model). Additionally, we could also flag *future* activity that is similar to a particular example or that comes from a specific cluster in our model. For instance, an operator might decide that a particular class of activities from the model, such as cars driving through a specified pedestrian zone, always (or never) warrants attention, simply by flagging the corresponding cluster in the scene model.

7.3 Assumptions and Limitations

In any intelligent system, simplifying assumptions must be made so that problems that would otherwise be AI-complete, i.e., require true intelligence, become computationally tractable. We devote a section to a summary of the assumptions made in our work because (1) such assumptions necessarily impose limits on the capabilities of the system and (2) relaxing the assumptions leads to intriguing further avenues of research.

First, several choices we make in this dissertation are motivated by the assumption that we are dealing with far-field scenes, i.e., scenes in which the moving objects are very small compared to their distance to the camera. In far-field scenes, articulated objects such as people appear as nearly rigid blobs, which makes it possible to use relatively simple descriptive attributes. Consequently, the system we have outlined in this document would not be able to spot activities that are unusual because of the way the moving object is articulated—e.g., a person dancing across the parking lot (unusual) versus a person simply walking across it

(typical). In a near-field setting, additional attributes such as the articulated pose of the object may have to be considered.

Second, an underlying assumption of our method is that anomalies are generally marked by a large deviation from the norm. However, as we mentioned in our discussion of missed detections, a person might consider an activity to be anomalous because of a small deviation that may be significant given additional information about the scene, such as the locations of sources, sinks and obstacles or a small deviation in an area where the object tracks in the same class tend to be particularly similar to one another. This suggests two intriguing avenues of further investigation: (1) incorporating semantic information into the anomaly detection framework and (2) considering the local variations of tracks belonging to a particular class when evaluating the likelihood of new examples.

Finally, in addition to considering examples of activities to be the paths of single moving objects through the scene, it would interesting to consider how our anomaly detection may be applied to interactions: co-occurences of multiple moving objects. One possibility would be to reason about the long-term co-occurence rates of activity classes as represented by the clusters in our learned scene model, essentially combining the techniques in this dissertation with those used for the co-occurence clustering of observations proposed by Stauffer [41].

7.4 Future Work

In addition to the general areas of further research outlined in the previous section, several open questions remain to be addressed:

- The size of the Nyström sample is important: if the sample is too small to be representative, the estimated spectral embedding may not be accurate, or even meaningful. Because we wanted to assess the performance of the statistical model for anomaly detection in our work, we chose the Nyström sample to be large. However, the computational complexity of classifying a new example is directly proportional to the size of the sample (we calculate pairwise comparisons between the new example and each of

the Nyström examples). Thus, the smaller the sample, the faster the classification. We are interested in exploring how the sample size affects the performance of our statistical anomaly detection.

- We have shown that incorporating additional object attributes into our distance measure makes the model richer and more informative. However, the number of classes (and thus the amount of data needed for training) is likely to grow with the number of attributes. It would interesting to investigate how this affects the size of the necessary Nyström sample and the size of the needed dataset.

- Scenes change over time, not only in appearance but also in the types of behavior that commonly occur. For instance, in our parking lot scene, a construction project might alter the paths of many objects in the scene and the changes might have long-term effects. Open questions remain in how such a statistical model can be adapted to long-term behavior changes—see Section 5.8 for a more detailed discussion.

Appendix A

Implementation

A.1 Pre-Processing of Tracks

In our data collection, we obtained moving object tracks by performing background subtraction and solving for the correspondences between connected foreground components in subsequent images using a constant size and velocity Kalman filter (see Chapter 2). Each track was a sequence of observation vectors $\boldsymbol{o}_i = [t_i, x_i, y_i, s_i]$, where t was the timestamp in miliseconds, (x, y) was the position of the centroid in image coordinates, and s was the area of the bounding box. The centroid coordinates x and y were normalized by the width and height of the image in pixels, respectively, to be between 0 and 1. The size s was normalized by the area of the image in pixels (i.e. a size of 1 would correspond to a bounding box covering the entire image). We applied a number of filtering steps to these original tracks:

1. We removed tracks for which the time interval between the first and last observation was less than 5 seconds.

2. For each track $T = \{\boldsymbol{o}_1, ...\boldsymbol{o}_n\}$, we measured the trajectory length $L(T)$ as $L(T) = \sum_{i=2}^{n} \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}$. We excluded tracks for which the trajectory length was less than 0.08. Extremely short tracks are not meaningful.

3. We excluded tracks for which the average observation size was larger than .02. Obser-

vations of that size tended to correspond to lighting effects such as cloud shadows and were larger than even the largest moving objects seen in the parking lot.

4. We excluded tracks in which the largest distance between any two consecutive centroid locations exceeded .2 (these were tracking errors where the tracker lost the object).

All of the above filtering steps were conservative in the sense that we picked thresholds for which we would not be filtering out any legitimate unusual activity, even at the cost of retaining a large proportion of tracking errors in the dataset.

We further pruned the filtered set of tracks so that each track contained observations at least 500ms aparts. We did this pruning by including the first obseration and then discarding any observations that occured less than 500ms after the last included observation. Finally, for each track we calculated the instantaneous velocity vectors $(\dot{x}_i, \dot{y}_i)$ by setting $\dot{x}_i = (x_i - x_{i-1})/(t_i - t_{i-1})$ and $\dot{y}_i = (y_i - y_{i-1})/(t_i - t_{i-1})$. The instantaneous velocity of the first observation was taken to be $(\dot{x}_1, \dot{y}_1) = (\dot{x}_2, \dot{y}_2)$.

Because both the velocity vectors and the bounding box sizes in a given track tend to be very noisy, we applied a moving average filter to both the instantaneous velocities as well as the sizes.

Appendix B

User Study Consent Form

CONSENT TO PARTICIPATE IN NON-BIOMEDICAL RESEARCH

Anomaly perception in far-field activity analysis

You are asked to participate in a research study conducted by Tomas Izo from the Department of Electrical Engineering and Computer Science at the Massachusetts Institute of Technology (M.I.T). The results of the study will contribute to Tomas' PhD thesis. You were selected as a possible participant in this study because of your affiliation with the computer vision and graphics community and your familiarity with the parking lot area between buildings 32 and 68. You should read the information below, and ask questions about anything you do not understand, before deciding whether or not to participate.

- **PARTICIPATION AND WITHDRAWAL**

Your participation in this study is completely voluntary and you are free to choose whether to be in it or not. If you choose to be in this study, you may subsequently withdraw from it at any time without penalty or consequences of any kind. The investigator may withdraw you from this research if circumstances arise which warrant doing so.

- **PURPOSE OF THE STUDY**

We wish to study how a person familiar with a particular scene—in our case, a view of the parking lot between buildings 32 and 68 from a camera on the 8th floor reading room of the Stata center—perceives the degree to which activity in the scene is unusual. We define "activity" as the movement of a single moving object, such as a car or a person, through the scene. The various aspects of an activity are the path that the object took through the scene, the size of object, its speed, direction of motion, etc. We have a statistical model that makes it possible to determine how *statistically* unusual a particular activity is and we wish to find out how the output of our system correlates with what a person familiar with the scene would perceive as unusual activity.

- **PROCEDURES**

If you volunteer to participate in this study, we will ask you to do the following things:

You will be presented with 60 to 100 examples of activity in the parking lot scene. Each activity will consist of the movement of a single object, such as a car or a person, through the scene. The examples will come from our database of recorded activity in the parking lot and will be shown as sequences of images of the object moving through the scene. In each image, the object will be shown superimposed on a faded static image of the scene. After the playback of each activity is complete, a summary image of the activity will be

shown. We will ask you to rate each example on the following scale based on how unusual it seems to you given your knowledge of the scene: very common, somewhat common, somewhat unusual or very unusual. Alternatively, because our data collection system sometimes makes errors and outputs sequences that link several different objects together, we will also give you the option to mark examples as erroneous, i.e. not corresponding to the movement of a single object. You can terminate the test at any time.

All testing will take place in an office in MIT building 32.

- **POTENTIAL RISKS AND DISCOMFORTS**

Each participant in the study will be interacting with a mouse and clicking approximately every five to sixty seconds over the course of a twenty- to thirty-minute session. If you have a prior history of a work-related repetitive stress injury, you may want to reconsider your participation in the study.

- **POTENTIAL BENEFITS**

It is not expected that you will receive any direct, personal benefits as a result of your participation in this study. However, the results will contribute to the general knowledge about how statistical scene and activity models compare with human perception of unusual or anomalous activity.

- **PAYMENT FOR PARTICIPATION**

No financial compensation will be offered in exchange for participation in this study.

- **CONFIDENTIALITY**

Any information that is obtained in connection with this study and that can be identified with you will remain confidential and will be disclosed only with your permission or as required by law.

The only identifiable information that will be included in this study are the participant's name and e-mail address. This information will be stored electronically and will be accessible only to the researchers who are directly involved in administering the study. No identifiable personal information will be included in any published results. Data will be electronically archived following the study. If other researchers use the data in future projects, personal identifiable information will be excluded.

- **IDENTIFICATION OF INVESTIGATORS**

If you have any questions or concerns about the research, please feel free to contact one of the following investigators:

Tomas Izo, Principal Investigator
Daytime phone: 617-258-5485
E-mail address: tomas@csail.mit.edu

Prof. Eric Grimson, Faculty Sponsor
Daytime phone: 617-253-4645
E-mail address: welg@csail.mit.edu

- **EMERGENCY CARE AND COMPENSATION FOR INJURY**

"In the unlikely event of physical injury resulting from participation in this research you may receive medical treatment from the M.I.T. Medical Department, including emergency treatment and follow-up care as needed. Your insurance carrier may be billed for the cost of such treatment. M.I.T. does not provide any other form of compensation for injury. Moreover, in either providing or making such medical care available it does not imply the injury is the fault of the investigator. Further information may be obtained by calling the MIT Insurance and Legal Affairs Office at 1-617-253 2822."

- **RIGHTS OF RESEARCH SUBJECTS**

You are not waiving any legal claims, rights or remedies because of your participation in this research study. If you feel you have been treated unfairly, or you have questions regarding your rights as a research subject, you may contact the Chairman of the Committee on the Use of Humans as Experimental Subjects, M.I.T., Room E25-143B, 77 Massachusetts Ave, Cambridge, MA 02139, phone 1-617-253 6787.

SIGNATURE OF RESEARCH SUBJECT OR LEGAL REPRESENTATIVE

I understand the procedures described above. My questions have been answered to my satisfaction, and I agree to participate in this study. I have been given a copy of this form.

_____
Name of Subject

_____
Name of Legal Representative (if applicable)

_____ _____
Signature of Subject or Legal Representative Date

SIGNATURE OF INVESTIGATOR

In my judgment the subject is voluntarily and knowingly giving informed consent and possesses the legal capacity to give informed consent to participate in this research study.

_____ _____
Signature of Investigator Date

Bibliography

[1] D. Arthur and S. Vassilvitskii. How slow is the k-means method? In *Proc. Symposium on Computational Geometry*, 2006.

[2] C.T.H. Baker. *The numerical treatment of integral equations*. Oxford: Clarendon Press, 1977.

[3] B.D.Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. International Joint Conferences on Artificial Intelligence*, pages 674–679, 1981.

[4] J. Bilmes. A gentle tutorial of the EM algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. Technical Report ICSI-TR-97-021, International Computer Science Institute, Berkeley, California, 1998.

[5] Dan Buzan, Stan Sclaroff, and George Kollios. Extraction and clustering of motion trajectories in video. In *Proc. IEEE International Conference on Pattern Recognition*, 2004.

[6] F.R.K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

[7] T.H. Cormen, C.E. Leiserson, and R.I. Rivest. *Introduction to Algorithms*. MIT Press, 1990.

[8] Hannah M. Dee and David Hogg. Is it interesting?: Comparing human and machine judgments on the pets dataset. In *Proc. IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS-ECCV04)*, 2004.

[9] Hannah M. Dee and David Hogg. Navigational strategies and surveillance. In *Proc. IEEE International Workshop on Visual Surveillance*, 2006.

[10] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *J. Royal Statist. Soc. Ser. B.*, 39(1), 1977.

[11] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*, section 3.9, Expectation-Maximization (EM), pages 124–128. Wiley-Interscience, New York, NY, second edition, 2001.

[12] Igor Fischer and Jan Poland. Amplifying the block-diagonal structure for spectral clustering. Technical Report IDSIA-03-05, Dalle Molle Institute for Artificial Intelligence, 2005.

[13] Charless Fowlkes, Serge Belongie, Fan Chung, and Jitendra Malik. Spectral grouping using the nystrom method. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 26(2), February 2004.

[14] Zhouyu Fu, Weiming Hu, and Tieniu Tan. Similarity based vehicle trajectory clustering and anomaly detection. In *Proc. IEEE International Conference on Image Processing*, 2005.

[15] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[16] Weiming Hu, Xuejuan Xiao, Zhouyu Fu, Dan Xie, Tieniu Tan, and Steve Maybank. A system for learning statistical motion patterns. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 28(9), September 2006.

[17] Weiming Hu, Dan Xie, Tieniu Tan, and Steve Maybank. Learning activity patterns using fuzzy self-organizing neural network. *IEEE Trans. Systems, Man, and Cybernetics, Part B: Cybernetics*, 34(3):1618–1626, 2004.

[18] L. Itti and P. Baldi. Bayesian surprise attracts human attention. In *Advances in Neural Information Processing Systems, Vol. 19 (NIPS*2005)*, pages 1–8, Cambridge, MA, 2006. MIT Press.

[19] Laurent Itti and Pierre Baldi. A principled approach to detecting surprising events in video. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition*, June 2005.

[20] Neil Johnson and David Hogg. Learning the distribution of object trajectories for event recognition. *Image and Vision Computing*, 14(8):609–615, 1996.

[21] Imran N. Junejo, Omar Javed, and Mubarak Shah. Multi feature path modeling for video surveillance. In *Proc. IEEE International Conference on Pattern Recognition*, 2004.

[22] S. Kang, J. Paik, A. Koschan, B. Abidi, and M. A. Abidi. Real-time video tracking using ptz cameras. *Proc. of SPIE 6th International Conference on Quality Control by Artificial Vision*, pages 103–111, May 2003.

[23] S. Lim, L. Davis, and A. Elgammal. A scalable image-based multi-camera visual surveillance system. In *AVSS*, pages 205–212, 2003.

[24] J. Migdal, T. Izo, and C. Stauffer. Moving object segmentation using super-resolution background models. In *Workshop on Omnidirectional Vision, Camera Networks and Non-classical cameras at ICCV*, October 2005.

[25] Joshua Migdal and W. Eric L. Grimson. Background subtraction using markov thresholds. In *IEEE Workshop on Motion and Video Computing*, January 2005.

[26] Andrew Naftel and Shehzad Khalid. Classifying spatiotemporal object trajectories using unsupervised learning in the coefficient feature space. In *Multimedia Systems*, 2006.

[27] Andrew Naftel and Shehzad Khalid. Motion trajectory learning in the dft-coefficient feature space. In *Proc. IEEE International Conference on Computer Vision Systems*, 2006.

[28] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and algorithm. In *Advances in Neural Information Processing Systems*, 2001.

[29] Lauren O'Donnell and Carl-Fredrik Westin. High-dimensional white matter atlas generation and group analysis. In *Proc. Medical Image Computing and Computer-Assisted Intervention*, 2006.

[30] P. Perona and W. T. Freeman. A factorization approach to grouping. Technical Report TR-99-03, Mitsubishi Electric Research Laboratory, 1999.

[31] Fatih Porikli. Learning object trajectory patterns by spectral clustering. In *Proc. IEEE International Conference on Multimedia and Expo*, 2004.

[32] Fatih Porikli and Tetsuji Haga. Event detection by eigenvector decomposition using object and frame features. In *IEEE Computer Vision and Pattern Recognition Workshop*, 2004.

[33] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical recipes in C*. 2nd ed. Cambridge University Press, 1992.

[34] M. Reiss and G. Taylor. Storing temporal sequences. *Neural Networks*, 4:773–787, 1991.

[35] A.W. Senior, A. Hampapur, and M. Lu. Acquiring multi-scale images by pan-tilt-zoom control and automatic multi-camera calibration. In *WACV05*, pages I: 433–438, 2005.

[36] Jianbo Shi and Jitendra Malik. Motion segmentation and tracking using normalized cuts. In *Proc. International Conference on Computer Vision*, 1998.

[37] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8), 2000.

[38] S. Sinha and M. Pollefeys. Towards calibrating a pan-tilt-zoom camera network. In *Workshop on Omnidirectional Vision and Camera Networks at ECCV*, 2004.

[39] C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Ft. Collins, CO, June 1999.

[40] C. Stauffer and K. Tieu. Automated multi-camera planar tracking correspondence modeling. In *CVPR*, July 2003.

[41] Chris Stauffer. Automatic hierarchical classification using time-based co-occurrences. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 1999.

[42] Chris Stauffer and W. Eric L. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):747–757, 2000.

[43] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet process. *Journal of the American Statistical Association*, 2006.

[44] H. Waessle and B.B. Boycott. Functional architecture of the mammalian retina. *Physiological Reviews*, 71:447–480, 1991.

[45] D. Wang and M. Arbib. Complex temporal sequence learning based on short-term memory. *Proc. of the IEEE*, 78(9):1536–1542, 1990.

[46] Xiaogang Wang, Xiaoxu Ma, and W. Eric L. Grimson. Unsupervised activity perception by hierarchical bayesian models. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2007.

[47] Xiaogang Wang, Kinh Tieu, and W. Eric L. Grimson. Learning semantic scene models by trajectory analysis. In *European Conference on Computer Vision*, 2006.

[48] Yair Weiss. Segmentation using eigenvectors: A unifying view. In *Proc. International Conference on Computer Vision*, pages 975–982, 1999.

[49] Eric W. Weisstein. ANOVA. From MathWorld–A Wolfram Web Resource, http://mathworld.wolfram.com/ANOVA.html.

[50] Eric W. Weisstein. p-value. From MathWorld–A Wolfram Web Resource, http://mathworld.wolfram.com/P-Value.html.

[51] Eric W. Weisstein. Spearman rank correlation coefficient. From MathWorld–A Wolfram Web Resource, http://mathworld.wolfram.com/SpearmanRankCorrelationCoefficient.html.

[52] Greg Welch and Gary Bishop. An introduction to the kalman filter. Technical Report TR 95-041, University of North Carolina at Chapel Hill, 1995.

[53] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: theory and its applications to image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15(11):1101–1113, 1993.

[54] L. Zelnik-Manor and M. Irani. Event-based analysis of video. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2001.

[55] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *Neural Information Processing Systems*, 2004.

[56] Zhang Zhang, Kaiqi Huang, and Tieniu Tan. Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes. In *Proc. IEEE International Conference on Pattern Recognition*, 2006.

[57] H. Zhong, J. Shi, and M. Visontai. Detecting unusual activity in video. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2004.

[58] X. Zhou, R. Collins, T. Kanade, and P. Metes. A master-slave system to acquire biometric imagery of humans at distance. In *ACM International Workshop on Video Surveillance*, November 2003.